AN ADAPTIVE BAG-OF-FEATURES FRAMEWORK FOR ARABIC HANDWRITING RECOGNITION

BY

MOHAMMED OMER HAJ ASSAYONY

A Dissertation Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

In

COMPUTER SCIENCE AND ENGINEERING

JANUARY 2017

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **MOHAMMED OMER HAJ ASSAYONY** under the direction his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE AND ENGINEERING

Mahmoud 22/1/2017

Dr. Sabri Abdullah Mahmoud
(Advisor)

Dr. Adel Fadhl Ahmed
Dean, College of Computer Science and Engineering

Dr. Shokri Z. Selim
(Member)

Alshayb

Dr. Mohammad R. Alshayeb
(Member)

Dr. Salam A. Zummo
Dean of Graduate Studies

Marwan Abu-Zaid

Dr. Marwan H. Abu-Amara
(Member)

24/1/17

Date

Dr. Dr. Lahouari Ghouthi
(Member)

To my parents, wife, kids, brothers and sisters

ACKNOWLEDGMENTS

First and foremost, I would like to thank Almighty Allah for blessing me with the will, patience, persistence and resources to accomplish this work.

My deepest appreciation and thanks are due to my advisor, *Prof. Dr. Sabri Abdullah Mahmoud* for his invaluable support and guide. He is always an excited supporter of my work, providing valuable ideas and advices. His help made this dissertation accomplishable. To him I shall forever remain thankful.

I am indebted to King Fahd University of Petroleum and Minerals for admitting me to the Ph.D. Program and providing the ideal environment for academic study and scientific research. I would also like to express deepest thanks to my dissertation committee members and to all instructors who taught me during the coursework.

My grateful thanks go to Hadhramout Establishment for Human Development, Yemen, for granting me scholarship to pursue my Ph.D. degree. Countless appreciations go to Alandalus University, Yemen, the institute where my skills and talents grew up. Special thanks go to *Dr. Salim A. Ramooda*, *Dr. Adnan A. Zain* and to *Mr. Saleh A. Lahmadi* who put my feet on the land of graduate studies.

I thank all my friends and colleagues who helped me from time to time. I am particularly grateful to *Irfan, Rashad, Dhiaa, Ahmed, Khalid, Fahd, Osama, Amer, Saeed, Sami, Abdullah, Abdurrahman* and *Mohammad.*

Last but certainly not the least, I wish to express my gratitude to my family. Special words of thanks are due to my parents, my brothers *Majdi & Mustafa* and all my family members for their encouragement. I am especially indebted to my wife *Um-Umar* for her tireless support throughout the entire period of my studies and her love has been a constant source of inspiration for me. Lastly, my loves go to my three awesome kids *Aishah*, *Umar* and *Ayah*.

Thanks so much to all of you

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

FULL NAME : MOHAMMED OMER HAJ ASSAYONY

THESIS TITLE : AN ADAPTIVE BAG-OF-FEATURES FRAMEWORK FOR ARABIC HANDWRITING RECOGNITION

MAJOR FIELD : COMPUTER SCIENCE AND ENGINEERING

DATE OF DEGREE : JANUARY 2017

Feature extraction is a crucial stage of pattern recognition systems. The advances in computer vision and machine learning have developed generic frameworks that could produce robust features for different domains. In this dissertation, we utilize the Bag-of-Features (BoF) framework for Arabic handwritten text recognition. We use the characteristics of handwritten text to improve the framework performance and enhance the quality of the produced features.

In this work, we have established a baseline for BoF framework that achieved state-of-the-art in recognizing isolated Arabic handwritten digits and subwords. Utilizing the characteristics of text images and handwritten text have significantly improved the framework computational performance. The framework is integrated with a handwriting recognition system based on Hidden Markov Models (HMMs). The first stage of the framework is adapted to the sliding window technique and the writing baseline of Arabic text is utilized for imposing localization in the produced features. The writing baseline has also inspired us to utilize multi-stream HMMs in a novel approach that significantly improved the recognition accuracy. In including the above enhancements, the recognition system achieved character recognition accuracy of 64.30% on KHATT database which is

better than the published accuracies on the same database using using traditional statistical features.

For the sake of comparison, we have implemented a handwriting recognition system based on Bernoulli Hidden Markov Models (BHMM) in which binary image representations are used as features. Our implementation has achieved character recognition accuracy of 63.28% on KHATT database which is comparable to the results we have achieved using the BoF with the traditional HMMs. We have proposed two approaches in order to reduce the dimensionality of the binary observations and to impose spatial localization. The two approaches achieved comparable recognition accuracies, in addition to the computational efficiency they have shown.

This work indicates that exploiting the context and the characteristics of Arabic handwritten text images as well as the careful adaptation of the framework improved the recognition accuracies.

ملخص الرسالة

الاسم الكامل: محمد عمر حاج السيـوني
عنوان الرسالة: هيكلية حقيبة السمات الملائم للتعرف الآلي على الكتابة العربية اليدوية
التخصص: علوم وهندسة الحاسب الآلي
تاريخ الدرجة العلمية: يناير 2017

يعتبر استخلاص السمات مرحلة مهمة في نظم التعرف على الأنماط. لقد أدى التقدم العلمي في مجال الرؤية بالحاسوب وتعليم الآلة إلى تطوير نماذج عامة لديها القدرة على استخلاص سمات قوية لمجالات مختلفة. نستغل في هذه الأطروحة نموذج حقيبة السمات في مجال التعرف الآلي على الكتابة العربية اليدوية ، حيث أننا نستخدم خصائص الكتابة اليدوية في تحسين أداء النموذج وكذلك في تحسين نوعية السمات المستخلصة.

لقد قمنا بإنشاء نوذج أساسي لحقيبة السمات استطاع أن يحقق نتائج قياسية في التعرف على الأرقام والكلمات العربية المكتوبة يدوياً. كما أدى استغلالنا لخصائص الصور المتضمنة للنصوص وخصائص الكتابة اليدوية إلى تحسين أداء النموذج بشكل ملحوظ.

بعد ذلك قمنا بربط النموذج مع نظام آلي للتعرف على الكتابة اليدوية مبني على نماذج ماركوف الخفية. خلال ذلك تم توليف المرحلة الأولى من النموذج لتتلاءم مع تقنية النوافذ المنزلقة، كما تم الاستفادة من خط الكتابة العربية الافتراضي في فرض التموقع في السمات المستخلصة، وفي استخدام نماذج ماركوف الخفية متعددة السبل بطريقة جديدة كان لها أثر ملحوظ في تحسين دقة التعرف.بتلك التحسينات مجتمعة تمكن النظام من تحقيق دقة تعرُّف بلغت 64.30% على مستوى الحروف في قاعدة البيانات (خط)، وهي أفضل من النتائج المنشورة على نفس قاعدة البيانات باستخدام السمات الإحصائية التقليدية.

من أجل المقارنة، فقد أنشأنا نظام تعرف آخر باستخدام نماذج ماركوف الخفية المبنية على توزيعات برنوللي، والذي يمتاز بقدرته على استخدام التمثيل الثنائي للصور كسمات. تمكن النظام من تحقيق دقة تعرف بلغت 63.28% على مستوى الحروف في قاعدة البيانات (خط)، وهي مقاربة للنتائج التي حققناها باستخدام نموذج حقيبة السمات مع نماذج

ماركوف الخفية التقليدية. ثم اقترحنا أسلوبين لكي نقلص من أبعاد المتجهات الثنائية ولنفرض تموقع مكاني ضمنها. كلا الأسلوبين حقق نتائج مقاربة، إضافة إلى الكفاءة الحاسوبية التي أظهراها.

يؤكد هذا العمل على أن استغلال سياق النص وخصائص الكتابة اليدوية وكذلك التطويع الدقيق للنموذج تودي إلى تحسين دقة التعرف على الكتابة العربية اليدوية.

CHAPTER 1

INTRODUCTION

This chapter provides an introductory material on Arabic handwritten text recognition. Section 1.1 highlights the importance of features in pattern recognition systems and the several approaches proposed for extracting features for the problem-at-hand. Section 1.2 provides a brief introduction to handwriting recognition problem and the processing phases involved in handwriting recognition systems. Section 1.3 presents the motivation of this study. Section 1.4 states the problem statement of the dissertation. Section 1.5 presents the research methodology that was followed to achieve the dissertation objectives. Section 1.6 highlights the contributions of the dissertation in the field of feature learning and handwriting recognition.

1.1. Feature Extraction and Learning

Applications of pattern recognition span several fields, including image categorization, visual object recognition (Zhang et al. 2007), speech recognition (Grosse et al. 2012) (Deng et al. 2010), text recognition (AbdulKader 2008) (Maalej & Kherallah 2016), word spotting (Aldavert et al. 2015) (En et al. 2016), writer identification (Christlein et al. 2015) …….. etc. The ultimate goal in these fields is to *learn* a model that can assign a category label to an input pattern. (Duda et al. 2001). In visual object recognition, the learned model would assign a name to an image of a real-world object. In text recognition, the learned model

would produce a transcription for the text shown in a text image. The typical approach to *learn* such a model is to provide a set of samples with their corresponding labels and *train* the model on them. Once trained, the model would predict labels for unseen samples. To achieve this goal, several processing stages are carried out. Figure 1.1 shows a general prototype for pattern recognition systems (Duda et al. 2001). The system takes in raw data representation of the real-world pattern that was captured by a proper sensor device such as a camera, scanner or microphone. The raw representation might pass through several preprocessing steps in order to enhance its quality and prepare it for the next phases. In visual object recognition systems, common preprocessing techniques/operations include noise removal, contrast adjustment and foreground/background segmentation. The feature extraction phase aims to enhance the pattern representation by extracting prominent attributes '*features*' that indicate the important characteristics of the pattern. The classification phase uses the extracted features to evaluate the learned model. Based on the evaluation, a category label is assigned to the pattern. The classification decision might be refined in order to improve the recognition performance. This refinement is accomplished in the post-processing phase. In handwriting recognition, the produced transcription might be evaluated against language models or spell checking systems to enhance the recognition accuracy.

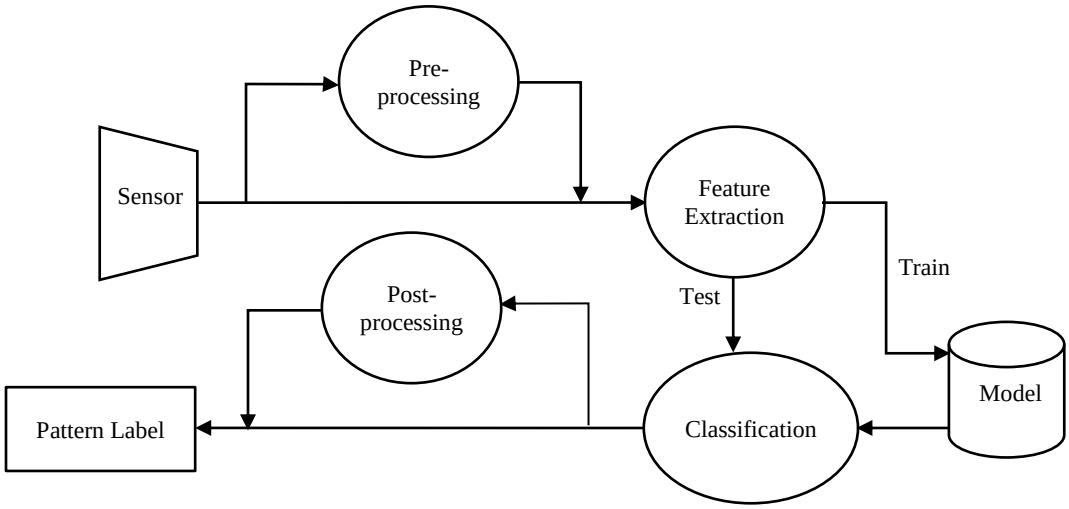


Figure 1.1: General prototype for pattern recognition system

The feature extraction phase is crucial as it simplifies the task of the classifier and improves its performance. However, deciding what type of features is suitable for a given domain is hard and requires considerable effort. Typically, features are designed for each domain based on characteristics of the domain. The main shortcoming in these *handcrafted features* is that they are very specific to the domain for which they were invented and it is hard to adapt them to other (even) similar domains. For instance, the features designed for human faces (Brunelli & Poggio 1993) might not be suitable for visual objects. Similarly, features designed for Latin text (Elms & Illingworth 1995) might not be suitable for Arabic text and vice versa (Abuhaiba et al. 1994) (Alma'adeed et al. 2004) (Haboubi et al. 2009).

The advances in computer vision and machine learning have led to the development of generic frameworks that produce robust features for different domains. These frameworks can be broadly divided into two categories based on the scope of their application. The first category includes frameworks that are applicable to domains that have similar topology, e.g., images with 2-D topology. SIFT descriptor (Lowe 2004) and the lower layers of Neocognitron (Fukushima 1980), Cresceptron (Weng et al. 1992) and HMAX (Poggio & Riesenhuber 1999) are examples of frameworks that would produce robust features for vision applications. These frameworks exploit the fundamental characteristics common among the domains to define the feature space. The abovementioned examples exploit the 2-D image structure in defining features based on the visual appearance of the image content.

The second category includes frameworks that are more generic and applicable to diverse domains, e.g., any high-dimensional representation of images, speech or natural languages.

PCA (Jolliffe 2002) (Tipping & Bishop 1999), sparse coding (Coates & Ng 2011b) (Grosse et al. 2012) and auto-encoders (Hinton & Zemel 1994) (Deng et al. 2010) are typical examples of this category. These frameworks aim to *learn* the correlations in data representation and extract the features based on the learned relations. The main property of these frameworks is that they are *trainable*, i.e., they are trained on samples from a specific domain in order to *learn* robust *representations* for that domain. The learned representations are used for defining the features for the domain samples. These frameworks are commonly known as *feature learning* or *representation learning frameworks*.

The *Bag-of-Features framework* (BoF) is an instance of these frameworks that was introduced in compute vision for image classification and retrieval (O'Hara & Draper 2011). The framework produces a *global feature vector* for an image by aggregating a set of *local features* extracted from the image. The *local features* are invariant to image transformations and deformations. Particularly, *local descriptors* that are based on the visual appearance of the image make the framework applicable to several image domains, including human faces, visual objects, natural scenes and handwritten text images.

The conversion of the local features into the global representation involves two phases, viz. encoding and pooling. In the encoding phase, the local features are transformed into another domain based on a predefined codebook that is learned in unsupervised manner from the local features of the training samples. The pooling phase aggregates the encoded features into a robust global representation. The learned codebook enables the encoding phase to produce domain-specific features from the local features. The pooling phase

makes the global representation invariant to spatial translation. BoF framework was applied to several applications, including the image classification and retrieval (Zhang et al. 2007) (Philbin et al. 2007), handwriting recognition (Rothacker et al. 2012), word spotting (Rothacker & Fink 2015) (Aldavert et al. 2015) and writer identification and verification (Fiel & Sablatnig 2013) (Christlein et al. 2015).

In this dissertation, we utilize BoF framework to produce robust features for handwritten Arabic text recognition. Though the framework was applied before in handwriting recognition (Rothacker et al. 2012), the applied framework was identical to what was used for visual objects and scene images. The properties of text images and the characteristics of Arabic handwritten text were not utilized. Our aim in this dissertation is to adapt BoF framework to Arabic handwriting recognition.

1.2. Handwriting Recognition

Handwriting recognition is a branch of pattern recognition concerned with the conversion of handwritten text images into editable text representation. It is an active research area as numerous research papers are annually published in journals and conference proceedings. The advances in handwriting recognition have assisted the automation of several demanding tasks in daily life, like bank checking processing, postal address reading and handwritten forms processing. Despite these successes, handwriting recognition still remains an open research problem (Parvez & Mahmoud 2013). The success in developing sophisticated systems for the abovementioned tasks is due to the restrictions applied to the task, as all the tasks have narrow domains with limited vocabularies and task-specific knowledge and constraints. In postal address reading, for example, the possible valid

vocabularies are limited to street and city names. Therefore, a dictionary of valid vocabulary can be built and utilized in the recognition. The recognition of open-vocabulary unconstrained handwritten text remains a challenging task. Besides the unexpected vocabularies, the immense variability in human writing style produces visual differences in size, slant and pen-stroke of characters' shapes within a sentence. Figure 1.2 shows two images of a single sentence each was written by two different writers.

Handwriting recognition is divided into two types: online and offline. In online handwriting recognition, the handwritten text is converted into digital representation in real-time. The writer uses a touch-sensitive device like a tablet, flat display or Personal Digital Assistant (PDA) for writing. At writing time, the device captures writing information like spatial location (x-y coordinates of the pen tip), temporal information and pen pressure. Offline handwriting recognition deals with text produced by pen and paper. To convert such text into digital representation, an image of the paper is acquired using scanner or camera and then text features are extracted and used for recognition. In this dissertation, we are addressing offline Arabic handwritten recognition. So throughout the text, the terms "*handwriting recognition*" and "*handwritten text recognition*" refer to the offline case.

The handwriting recognition system has a structure similar to the one shown in Figure 1.1. The preprocessing phase improves the quality and appearance of the text images. Noise removal, binarization, skew/slant correction and text-baseline detection are examples of the preprocessing tasks. Further, in this phase the text line image might be segmented into words, characters, strokes or other units in order to recognize each unit separately. Some

systems require a text image to be converted into more concise representation such as thinning (skeletonization) or contour representations.

The feature extraction phase extracts relevant attributes for recognition. The traditional features used in handwriting recognition can be divided into two types: structural and statistical features. Structural features are properties describing character shapes and writing style, such as loops, branch-points, endpoints, and dots. The statistical features are numerical measures computed over images or regions of images. Such features include pixel densities (Märgner et al. 2006), wavelet response (Bhattacharya & Chaudhuri 2003) (Fink & Plotz 2005), and discrete cosine transform coefficients (AlKhateeb et al. 2008). The features extracted from the training samples are used to build models that can be used to recognize novel samples. Once the model is built, the recognition of novel samples can be considered as classification and off-the-shelf classification techniques can be exploited. Researchers have exploited different classification techniques including k-Nearest Neighbors (kNN), Artificial Neural Networks (ANN), Support Vector Machines (SVM), Hidden Markov Model (HMM), and Recurrent Neural Networks (RNN). The post-processing phase utilizes contextual information, language models or spell checking systems to improve the results of the recognition phase.

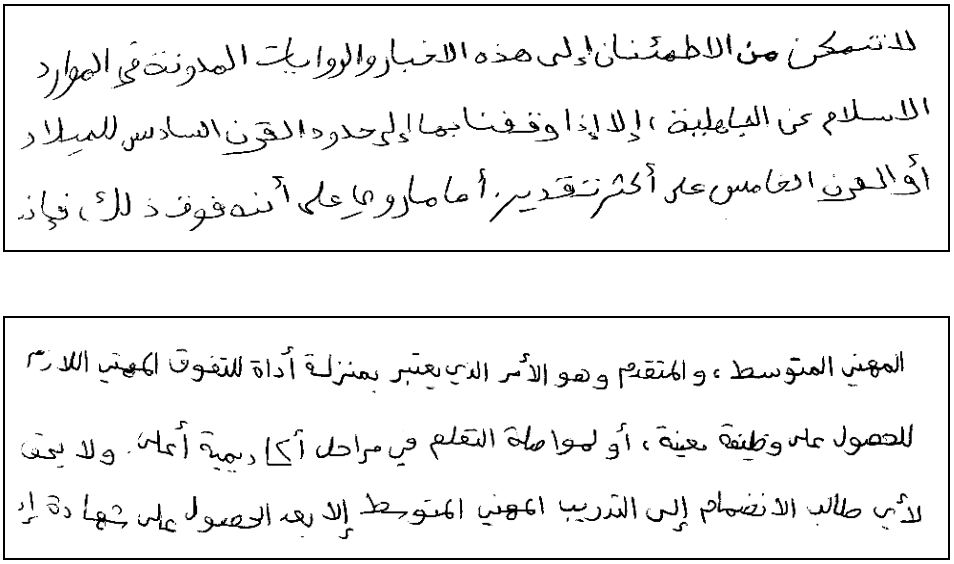


Figure 1.2: Two sentences written by two different writers from KHATT database (Mahmoud et al. 2014)

1.3. Motivation

Handwriting recognition is an active research area and it is applicable in several daily life applications. Handwriting recognition system comprises multiple phases where each phase contributes one way or another in the performance of the whole system. At the heart of the recognition system is the feature extraction phase that aims to deliver relevant attributes to the recognizer. In traditional systems, features are handcrafted based on experts' knowledge and experience. This process is tedious and time-consuming. Furthermore, handcrafted features are domain-specific and it is hard to adapt them to other domains. Features designed for Latin text might not be suitable for Arabic text and those designed for machine-printed text might not be suitable for handwritten text. The advances in computer vision and machine learning has led to the development of generic frameworks that would produce robust features for different domains. Several research groups have utilized different frameworks in handwriting recognition with remarkable achievements, particularly in the recognition of isolated digits and words (Cireşan, U. Meier, et al. 2012) (Graves & Schmidhuber 2009). BoF framework has been utilized in many document image analysis applications, including handwriting recognition (Rothacker et al. 2012). The application of BoF framework to handwriting recognition was identical to what was used for visual objects and scene images without utilizing the properties of text images and handwritten text. As BoF is an instance of unsupervised feature learning, one of the main characteristics of these framework is the ease to embed domain knowledge to enhance the quality of the produced features (Deng & Yu 2014). This motivates us to adapt BoF framework to Arabic handwriting recognition and augment it with the context and utilize the characteristics of Arabic text. We are unaware of any research applying feature learning

techniques in the recognition of open-vocabulary unconstrained Arabic handwritten text. Most of the published works on the recognition of Arabic handwritten text were applied to closed-vocabulary problems using IfN/ENIT database (Pechwitz et al. 2002). KHATT database (Mahmoud et al. 2014) is an open-vocabulary database of natural unconstrained Arabic handwritten text that we will use in this work.

1.4. Problem Statement

The problem of this dissertation can be stated as follows:

Given an open-vocabulary database of unconstrained Arabic handwritten text lines' images, how can we adapt the Bag-of-Features framework to extract robust features that can be integrated into a handwriting recognition system to achieve high character recognition rates? How can we utilize the context and Arabic text characteristics to address the limitations of the Bag-of-Features framework and enhance the quality of the produced features?

1.5. Research Methodology

The main objective of this dissertation is to conduct basic and applied research in handwriting recognition of open-vocabulary unconstrained Arabic handwritten text. The context and characteristics Arabic handwritten text were utilized in adapting BoF framework to the handwriting recognition.

The following methodology is followed to carry out the main phases of the dissertation in order to achieve the dissertation objective.

1. Extending the literature review of the state-of-the-art approaches related to feature learning and their applications related to the dissertation.
2. Establishing a baseline for feature learning system based on BoF framework.
3. Evaluating the established baseline on holistic handwriting recognition.
4. Integrate the framework baseline with a segmentation-free handwriting recognition system and evaluate it on open-vocabulary off-line Arabic handwritten text recognition.
5. Enhance the baseline by utilizing the context and characteristics of Arabic handwritten text.

1.6. Contributions of the Dissertation

This dissertation addressed the adaptation of BoF framework to the handwriting recognition systems of Arabic text. The contributions of this dissertation to the field of feature learning and handwriting recognition are summarized in the following points:

1. We established a baseline of BoF framework for Arabic handwriting recognition. As the framework involves several steps that can be implemented by a variety of techniques, we thoroughly investigated several techniques, focusing on the options and parameters that have impact on the quality of the produced features. We integrated the framework baseline with a holistic handwriting recognition system and evaluated it against two Arabic handwritten text datasets, the non-touching Arabic Indian digits and the Arabic sub-words datasets of CENPARMI Bank check database (Al-Ohali et al. 2004). The recognition system achieved a recognition

accuracy of 99.34% and 89.93% on the two datasets, respectively. Both results outperformed the state-of-the-art accuracies reported in the literature (Section 3.2).

2. The utilization of the characteristics of the handwritten text led us to propose two novel versions of SIFT that achieve the discriminative power of SIFT and are computationally efficient with half the size the SIFT descriptors. The two versions achieved comparable recognition performance to the original SIFT on the abovementioned datasets, in addition to the recognition accuracies, the two versions take less time to compute, and due to their lower dimensionality, the clustering and quantization phases became computationally efficient (Section 3.3).
3. We investigated integrating the learning stage of BoF framework with Gabor filter response features that previously achieved prominent performance on handwriting recognition. Gabor filter response features were arranged into two layouts and replaced SIFT descriptors. Though the recognition accuracies of the produced features were lower, the experiments showed that adapting low-level features to the format of local descriptors would significantly improve their discriminative power. (Section 3.4).
4. We integrated the framework baseline with a segmentation-free handwriting recognition system based on Hidden Markov Models (HMMs). We adapted the local features extraction stage of BoF framework to the nature of the sliding window strategy. Then, we utilized the characteristics of the Arabic handwritten text for imposing spatial localization in BoF representation. This adaptation motivated us to utilize multi-stream HMMs in a novel style that significantly improved the performance. The recognition system was evaluated against KHATT

database, the public open-vocabulary database of natural unconstrained Arabic handwritten text (Mahmoud et al. 2014). The system achieved character recognition accuracy of 64.30% which is prominent achievement on such a challenge dataset. The same recognition system with traditional statistical features achieved 46.13% character recognition accuracy on the same dataset (Mahmoud et al. 2014) (Chapter 4).

5. We implemented a segmentation-free recognition system based on Bernoulli Hidden Markov Models (BHMMs). This system deals directly with the raw binary representation of the text images, so it eliminates the feature extraction phase of the recognition system. The system achieved character recognition accuracy of 63.28% on KHATT database which is comparable to the results we achieved with BoF representation with the traditional HMMs system using the BoF representations. In addition, we proposed two approaches, coined as the local sampling and the local cell layers, in order to enhance the quality of the binary observations produced by the sliding window strategy. The two approaches achieved character recognition accuracy rate of 63.34% and 61.56%, respectively. The two approaches significantly speedup the computation of the system training and evaluation (Chapter 5).

The remaining of this dissertation is organized as following. In Chapter 2, we present the foundations of feature learning and brief introductory to the techniques that are used throughout the dissertation. In Chapter 3, we present the baseline of the BoF framework, the presentation and evaluation of the two proposed versions of SIFT, and the utilization of the Gabor filter response features with BoF framework. In Chapter 4, we present the

adaptation of the framework baseline to a segmentation-free HMM-based handwriting recognition system and the utilization of the characteristics of Arabic handwritten text for imposing spatial localization in BoF representation. Then we present and analyze the results of the experiments we carried out on KHATT database. In Chapter 5, we present our implementation of the segmentation-free BHMM-based recognition system and the two approaches we proposed for enhancing the quality of the binary observations produced by the sliding window strategy. This is followed by the results of the thorough evaluation on KHATT database. In Chapter 6, we present our conclusions and directions for future extensions.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

This chapter presents the foundations of feature learning and of the techniques and algorithms that are used throughout the dissertation. It also reviews the literature of feature learning and its applications to handwriting recognition and document image analysis. Section 2.1 presents an abstract architecture from which the different feature learning frameworks can be instantiated. Section 2.2 reviews the feature learning frameworks that have been applied to handwriting recognition. As our aim in this dissertation is to adapt the Bog-of-Features (BoF) framework to handwriting recognition, we present in-depth review of BoF framework and its applications to handwriting recognition and document image analysis in Sections 2.3 and 2.4, respectively. In Section 2.5, we present a brief background of the algorithms that are used in implementing BoF framework. Section 2.6 presents a brief background of Gabor filter response features that are utilized in implementing the first stage of BoF framework. Section 2.7 draws the conclusions of the chapter.

2.1. General Architecture for Feature Learning Frameworks

Learning robust representations from low-level data representation is an active topic in machine learning and computer vision. large number of frameworks have been invented that are drastically different in their design, structure (single-stage vs. multi-stage), training approach (supervised vs. unsupervised) and the domain of training samples (labeled samples, unlabeled samples, and out-of-domain samples). Nevertheless, feature learning

frameworks that were developed for vision applications possess generic architecture, inspired by Hubel and Wiesel biological study (Hubel & Wiesel 1962) of the cat's primary visual cortex in the sixties of the last century (LeCun 2012). The study showed that *the primary visual cortex* consists of two different types of cells: *simple-cells* and *complex-cells*. Simple cells respond to certain properties of the input pattern such as edge structures and their orientation, and they are very sensitive to the location and orientation of the sensed elements. Complex cells aggregate the response of several adjacent simple cells and pass them to the upper stages. Unlike simple cells, complex cells are less sensitive to the location and they show degree of invariance to small shifts. The visual cortex system involves several stages of the simple-complex cell architecture with complex bidirectional connections to enable the system detecting higher-level representation for the visual scenes. The abstract architecture of the feature learning framework that simulate a single-stage simple-cells-complex-cells visual cortex is shown in Figure 2.1 (LeCun 2012).

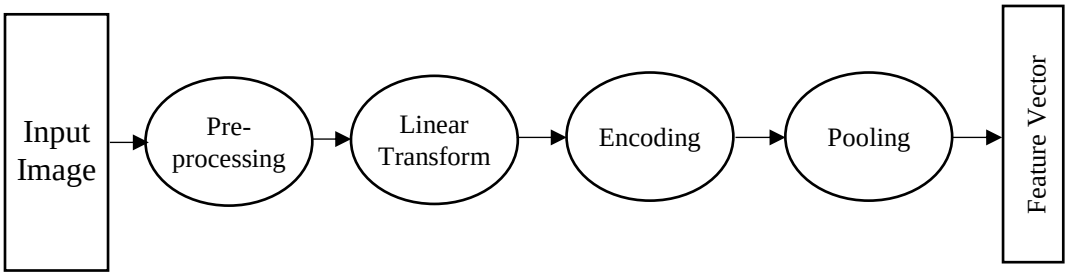


Figure 2.1: The general architecture of the single-stage feature learning framework

The input is an image represented by raw pixel intensities. The aim of the preprocessing layer is to improve the quality of the input representation and remove the noise associated with it. Smoothing, normalization, de-correlation and whitening are examples of image preprocessing tasks. The linear transformation layer applies linear operations like matrix-matrix additions and multiplications or spatial convolution on the preprocessed input. The purpose of this layer is to highlight the salient features of the input pattern. The main property of the feature learning frameworks is that the basis of the linear operation (the matrix elements or the elements of the convolution filters) are *learned* from the low-level data representation. This property makes these framework generic and applicable to different domains without the need for human intervention. The basis are learned by training the framework either in supervised (using labeled samples) or unsupervised (using unlabeled samples) manner. The encoding layer applies a predefined non-linear transformation like quantization, tanh, softmax and sparsification. This layer transforms the input to another domain in which semantically different patterns can be easily differentiated. In the pooling layer, the adjacent encoded elements are aggregated in order to make the final representation invariant to distortion and small sifts in the input. It could also reduce the dimensionality of the final representation by combining similar and adjacent elements into a single representation. Common aggregation functions are the sum, average, max, L_p-norm and log-mixture.

The linear transform and encoding layers of the framework together simulate the behavior of the simple cells of the primary visual cortex, while the pooling layer simulates the behavior of the complex cells. The multi-stage feature learning frameworks are constructed by stacking up several stages of the 4-layer architecture in a pipeline approach such that a

stage output is fed as input to the next upper stage. The feedback connections involved in the visual cortex system are eliminated by these computational systems in order to simplify the computational processing. Figure 2.2 depicts an abstraction of N-stages feature learning frameworks. Each stage possesses the 4-layers architecture shown in Figure 2.1.

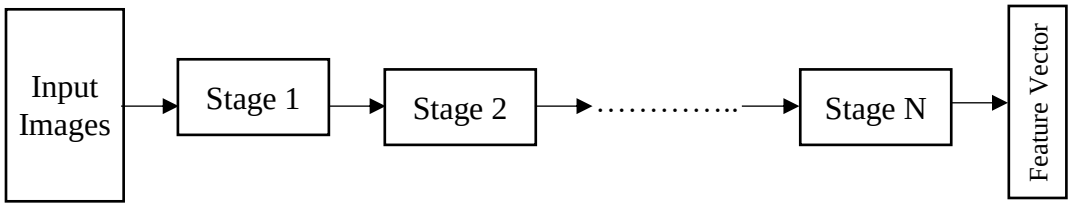


Figure 2.2: The multi-stage feature learning framework

2.1.1. Handcrafted Feature Extraction vs. Feature Learning Frameworks

The popular feature extraction algorithms in computer vision known as *local descriptors*, *e.g.*, SIFT (Lowe 2004), SURF (Bay et al. 2008), CS-LBP (Heikkilä et al. 2009) and BILD (Zhang et al. 2014), were also inspired by Hubel and Wiesel biological model and have architecture similar to the one shown in Figure 2.1 (Brown et al. 2011) (LeCun 2012). For instance, SIFT applies Gaussian smoothing in the preprocessing layer. The linear transform is achieved by convolving the image (or an image patch) with gradient filters. The encoding function distributes the gradient magnitude of image pixels between the two closest orientation bins. The pooling layer is implemented by summing up the gradient magnitude in each orientation bin within 4×4 spatial sub-regions. SURF algorithm (Bay et al. 2008) is an approximation of SIFT designed for computational efficiency. It approximates the gradient computations by Haar wavelet responses computed efficiently using integral images. The Biologically Inspired Local Descriptor (BILD) (Zhang et al. 2014) uses Gabor filters in the linear transform stage instead of the gradient filters used by SIFT. The CS-LBP algorithm (Heikkilä et al. 2009) is another algorithm inspired by SIFT. Instead of using the gradient filters in the linear transform stage, CS-LBP utilized the Local Binary Pattern (LBP) (Ojala et al. 2002), the texture operator that achieved remarkable performance in face recognition, as linear transformation operation.

The main difference between these algorithms and the feature learning frameworks is that the basis for the linear transform layer are predefined. This explains why these features are occasionally named *handcrafted features* (Jarrett et al. 2009) (LeCun 2012). These algorithms played crucial rule in the design of efficient feature learning frameworks of low complexity that achieved satisfactory performance with reasonable training samples.

2.1.2. Supervised vs. Unsupervised Feature Learning Frameworks

The feature learning frameworks can be classified based on the training approach into supervised and unsupervised frameworks (Coates 2012) (Deng & Yu 2014). In *supervised frameworks*, the basis of the linear transform layers of all stages are trained in supervised manner, together with the classifier's parameters, using *labeled samples*. In unsupervised frameworks, however, the basis are learned offline using *unlabeled samples*. Once they have been learned, the framework is applied to extract features for input samples (in training and testing sets). The extracted features could be utilized by traditional off-the-shelf recognizers and classifiers. The two learning approaches can be combined to improve the performance of each other, giving *the hybrid frameworks* (Deng & Yu 2014). A supervised framework (e.g., CNNs) can be initialized by basis that have been learned using unsupervised approach in order to avoid local minima and to alleviate the need for large set of labeled samples for training (Erhan et al. 2010). On the other hand, the unsupervised framework (e.g., DBNs) that has been pre-trained by unlabeled samples can be further tuned using labeled samples to adapt the whole system to the target domain (Salakhutdinov & Hinton 2007).

Each category has its own advantages and disadvantages. Supervised frameworks are more generic and can be easily adapted to new domains. The main shortcoming is that they require abundant labeled samples for training. In (Goodfellow et al. 2016) the authors stated "*As of 2016, a rough rule of thumb is that a supervised deep learning algorithm will generally achieve acceptable performance with around 5,000 labeled examples per category, and will match or exceed human performance when trained with a dataset containing at least 10 million labeled examples."* While it could be possible to acquire

such huge dataset for visual objects or natural scenes, it seems unfeasible for other domains like handwritten text. On the other hand, the unsupervised frameworks are flexible, so they can be easily adjusted based on the specific domain knowledge (Deng & Yu 2014). For handwriting recognition, this property could be exploited to enhance the performance of these frameworks by augmenting them by the context and the characteristics of text images and handwritten text.

It is worthily mentioning here that because the basis of unsupervised frameworks are learned offline, some of these frameworks have slightly different architecture than that depicted in Figure 2.1. Most of unsupervised frameworks presented in Section 2.2 apply the identity transformation ($f(x) = x$). Indeed, the basis are utilized in the encoding layer. For instance, in the single-stage feature learning framework presented in (Coates, Lee, et al. 2011) and the second stage of BoF frameworks, the non-linear transform (vector quantization) utilizes the basis (the codebook) that have been learned by the unsupervised learning algorithm (k-means clustering) in the transformation.

2.1.3. Semi-Supervised Learning, Self-Training, Transfer Learning and Self-Taught Learning

To cope with the scarce of training samples, several learning approaches were proposed. *Semi-supervised learning* utilizes both labeled and unlabeled samples in training supervised frameworks. According to (Kingma et al. 2014), the semi-supervised learning is defined as the answer of the question of "*how can properties of the data be used to improve decision boundaries and to allow for classification that is more accurate than that based on classifiers constructed using the labelled data alone?*". *Self-training* is the simplest paradigm of the Semi-supervised learning (Kingma et al. 2014). In self-training,

the supervised framework is trained with the available labeled samples. Then, the trained framework is used to predict labels for the unlabeled samples. Those samples whose labels were predicted with high confidence are utilized in retraining the framework. This paradigm was utilized in (Frinken & Bunke 2010) to improve the performance of RNN-based handwriting recognition system for Latin script.

Transfer learning enables a supervised framework that was trained for one task to be used in another similar task. It is helpful in such situations where the first task has abundant training set but few training samples are available for the second task (Oquab et al. 2014) (Bengio et al. 2013) (Deng & Yu 2014). Transfer learning was utilized in (Cireşan, Ueli Meier, et al. 2012) to improve the performance of handwriting recognition systems. A CNN-based handwriting recognition system that was trained on isolated handwritten digits was utilized in recognizing uppercase Latin characters and the later was utilized in recognizing Chinese characters.

Self-taught learning is the transfer learning variant for unsupervised approach. Usually, unsupervised frameworks assume that the training unlabeled samples could be labeled but their target label class is unknown. Self-taught learning enables unsupervised frameworks to utilize unlabeled samples from different categories and different domains (Raina et al. 2007) (Le 2013). For visual object recognition, the unlabeled samples could be image patches cropped from images of random visual objects. For handwritten text, the unlabeled samples could be image patches cropped from handwritten text images. The framework utilizes these samples to learn primitive structures (e.g., edges, corners, curve…..) that are common in the domain. Self-taught learning has been utilized to improve the performance

of unsupervised feature learning frameworks applied to handwriting recognition, especially those for cursive text as it is hard to provide pre-segmented handwritten characters (Hammerla et al. 2010) (Rothacker et al. 2012).

2.2. Feature Learning Frameworks for Handwriting Recognition

In the last two decades, several feature learning frameworks were applied to learn robust representations for handwritten text. This section provides a comprehensive survey of the applications of feature learning frameworks to handwriting recognition. As the depth of the framework (number of the stacked stages) plays crucial rule in the quality of the produced features, the reviewed works are categorized based on the number of stages in the framework. We divide the feature learning frameworks into three categories: the single-stage, two-stage and multi-stage feature learning frameworks. The instances of each category that were applied to handwriting recognition are presented in the next subsections. Table 2.1 gives a summary for the works reviewed in the next subsection.

Table 2.1: Summary of feature learning frameworks applied to handwriting-related applications

Work	Work Description	Database	Framework	Notes
(Coates, Carpenter, et al. 2011)	Text detection and character recognition in scene images	ICDAR 2003	Single-stage unsupervised framework	Self-taught learning was applied
(Pechwitz & Maergner 2003) (Märgner et al. 2006)	Arabic handwriting recognition	IfN/ENIT	Loeve-Karhunen Transform	Dimensionality reduction technique
(Dreuw et al. 2008) (Dreuw et al. 2009)	Arabic handwriting recognition	IfN/ENIT	PCA	Dimensionality reduction technique
(Hamdani et al. 2013)	Arabic handwriting recognition	MADCAT KHATT	PCA	Dimensionality reduction technique
(Fink & Plotz 2005) (Kozielski et al. 2013) (Bluche et al. 2013b)	Latin handwriting recognition	IAM RIMES	PCA	Dimensionality reduction technique
(Bayat 2014)	Farsi digits recognition	Hoda	PCA	Dimensionality reduction technique
(Wang et al. 2012)	Text detection and character recognition in scene images	ICDAR 2003	Two-stage hybrid framework The first stage is unsupervised The second stage is supervised	Self-taught learning was applied in the first stage
(Al-dmour & Abuhelaleh 2016)	Arabic words recognition	IfN/ENIT	BoF	Self-taught learning was applied
(Rothacker et al. 2012)	Arabic handwriting recognition	IfN/ENIT	BoF	Self-taught learning was applied

(Cireşan et al. 2010)	Digit recognition	MNIST	DNN	
(Cireşan, U. Meier, et al. 2012)	Latin/Chinese character recognition	NIST SD 19 CASIA	DNN	Transfer learning was applied
(Cireşan & Schmidhuber 2013)	Chinese character recognition	HWDB 1.1	DNN	
(LeCun et al. 1989) (LeCun et al. 1990) (Simard et al. 2003) (Chellapilla, Shilman, et al. 2006) (Pan et al. 2015)	Digit recognition	MNIST	CNN	
(Chellapilla, Puri, et al. 2006) (Cireşan et al. 2011) (Liu et al. 2013)	Digit and Latin character recognition	MNIST NIST SD 19 Private database	CNN	
(LeCun et al. 1997)	Courtesy amount recognition	Private database	CNN	Trained on pre-segmented letters. Over-segmentation heuristics and implicit HMM-based segmentation were applied on the continuous text
(Bluche et al. 2013a)	Latin handwriting recognition	RIMES	CNN	Over-segmentation heuristics and implicit HMM-based segmentation were applied on the continuous text
(Elleuch et al. 2016)	Arabic character recognition	HACDB IfN/ENIT	CNN	Words from IfN/ENIT were pre-segmented
(AbdulKader 2008)	Arabic handwriting recognition	IfN/ENIT	CNN	Holistic approach at Part of Arabic Word (PAW) level

(Chellapilla & Simard 2006)	East-Asian character recognition	Private database	CNN	
(Soman et al. 2013) (Anil et al. 2015)	Machine-printed Malayalam character recognition	Private database	CNN	
(Kim & Xie 2015)	Hangul character recognition	PE92 SERI95a	CNN	
(Graves & Schmidhuber 2009) (Graves 2012) (Liwicki et al. 2012) (Maalej et al. 2016) (Maalej & Kherallah 2016)	Arabic handwriting recognition	IfN/ENIT	RNN	
(Pham et al. 2014)	Arabic/Latin handwriting recognition	OpenHaRT IAM RIMES	RNN	
(Bluche, Louradour, et al. 2014)	Latin handwriting recognition	IAM RIMES	RNN	
(Breuel et al. 2013)	Latin machine-printed text recognition	UW3 Private dataset	RNN	

(Hinton et al. 2006) (Salakhutdinov & Hinton 2007)	Digit recognition	MNIST	DBN	
(Porwal et al. 2012)	Arabic handwriting recognition	AMA	DBN	Holistic approach at Part of Arabic Word (PAW) level
(Elleuch et al. 2015a)	Arabic handwritten character and word recognition	HACDB IfN/ENIT	DBN	
(Elleuch et al. 2015b)	Arabic handwritten character recognition	HACDB	DBN	
(Hammerla et al. 2010)	Arabic/Latin handwriting recognition	IfN/ENIT IAM	DBN	Self-taught learning was applied

2.2.1. Single-Stage Feature Learning Frameworks

Several unsupervised feature learning frameworks applied to visual object classification comprise a single stage structure of Figure 2.1 (Raina et al. 2007) (Coates, Lee, et al. 2011) (Coates & Ng 2011b). The main advantage of such frameworks is that the training is relatively simple compared to frameworks with many stages. The trained single stage is able detect low-level representation of input patterns. For images, the single-stage frameworks can detect directed edges. The detected representations are encoded and pooled to build the final features for the whole image.

The single-stage framework presented in (Coates, Lee, et al. 2011) applies whitening in the preprocessing layer. The basis of the linear transform layer were learned by applying k-means clustering on a set of random decorrelated patches. Several non-linear transformations for the encoding layer are evaluated in order to quantify their performance with the learned basis. For pooling layer, the sum pooling was utilized. The framework was applied for detecting and recognizing machine-printed text in scene images in (Coates, Carpenter, et al. 2011).

The dimensionality reduction techniques, e.g., Principle Component Analysis (PCA) can be viewed as a single-stage unsupervised feature learning framework (Coates & Ng 2011b) (Bengio et al. 2013). These approaches de-correlate the feature vectors and transform them to lower dimensional space, so the pooling stage is absent. The dimensionality reduction approaches were applied to produce robust features for HMM-based handwritten recognition systems (Pechwitz & Maergner 2003) (Fink & Plotz 2005) (Märgner et al.

2006) (Dreuw et al. 2008) (Dreuw et al. 2009) (Hamdani et al. 2013) (Kozielski et al. 2013) (Bluche et al. 2013b) (Bayat 2014).

2.2.2. Two-Stage Feature Learning Frameworks

The feature learning framework presented in (Wang et al. 2012) added a supervised stage to the single-stage framework of (Coates, Lee, et al. 2011), giving a two-stages framework. While the first stage is trained offline in unsupervised manner, the second stage is trained with the classifier in supervised manner. The framework was applied to detect and recognize machine-printed text in scene images.

The popular *Bag-of-Features* framework (Csurka et al. 2004) and its extensions (Spatial Pyramid Matching (SPM) (Lazebnik et al. 2006), Fisher Vector (FV) (Perronnin et al. 2010) (Sánchez et al. 2013), Vector of Locally Aggregated Descriptors (VLAD) (Arandjelovic & Zisserman 2013) and the Super-Vector Coding (SVC) (Zhou et al. 2010)) have two-stage architecture. The first stage is implemented by local descriptor algorithms (e.g., SIFT) for extracting robust invariant features. The second stage is an *unsupervised feature learning framework* whose input is the features extracted in the first stage, i.e., the descriptor vectors. The basis of the second stage are usually learned by clustering algorithms (k-means clustering). The difference between these models are in the non-linear transform encoding. While the naïve BoF framework utilized the vector quantization encoding, the early SMP framework (Lazebnik et al. 2006) relied on soft-assignment approaches to avoid the quantization distortion associated with the vector quantization encoding. The FV and VLAD applied soft assignment based on Gaussian Mixture Models (GMMs) that were estimated over the training samples.

The two-stage frameworks exploit the low-level features of the first stage in producing higher features referred to as *mid-level representation*, as they can't provide representation for high-level structured image attributes (e.g., object parts) (Boureau, Bach, et al. 2010). Mid-level representations are descriptive statistics that can be utilized by off-the-shelf classifiers. The integration of mid-level features with supervised classifiers achieved state-of-the-art accuracies for visual object classification and recognition (Law et al. 2014). These features were also utilized in handwriting recognition (Rothacker et al. 2012), word spotting (Rothacker & Fink 2015) (Aldavert et al. 2015) (Shekhar & Jawahar 2013) (Rusiñol et al. 2011), writer identification (Christlein et al. 2015) (Fiel & Sablatnig 2013) and identifying machine-printed vs. handwritten text (Zagoris et al. 2014).

2.2.3. Multi-stage Feature Learning Frameworks

The earlier frameworks inspired by the Hubel and Wiesel model had multi-stage architecture. The Neocognitron (Fukushima 1980), Cresceptron (Weng et al. 1992) and HMAX (Poggio & Riesenhuber 1999) are examples of such frameworks. The basis of the linear transform layers are either fixed (handcrafted) or they are learned stage-wise in unsupervised manner.

The *hyperfeatures framework* (Agarwal & Triggs 2006) is the multi-stage variant of the Bag-of-Features framework. It is a hierarchical model that comprises a *local descriptor stage* followed by several stages of *unsupervised feature learning*. The basis of the linear transform layers are learned by clustering. The vector quantization and GMM-based soft assignment are utilized in the encoding layers. The sum operation is utilized in the pooling layers of the stages.

The Deep Fisher Network proposed in (Simonyan et al. 2013) is the multi-stage variant of the Fisher Vector framework that achieved comparable performance to the Deep Neural Networks in visual object classification and recognitions, yet it is computationally efficient.

The unsupervised feature learning framework presented in (Coates & Ng 2011a) is the multi-stage variant of (Coates, Lee, et al. 2011). The hierarchical architecture is similar to the hyperfeatures framework except for the first stage which is a trainable stage too.

The more sophisticated multi-stage feature learning frameworks are instances of deep neural networks (DNNs), the neural networks comprises multiple hidden layers (Bengio et al. 2013) (Deng 2014) (Deng & Yu 2014) (Schmidhuber 2015) (LeCun et al. 2015) (Goodfellow et al. 2016). The trained network would extract robust representation for the input samples, where upper layers extract more abstract feature representations.

The DNNs were applied to the recognition of isolated digits (Cireşan et al. 2010) (Cireşan, U. Meier, et al. 2012) and Chinese characters (Cireşan & Schmidhuber 2013). As reported in (Krizhevsky et al. 2012), the DNN presented in (Cireşan, U. Meier, et al. 2012) achieved the performance near to the human ability in recognition of handwritten MNIST numerals dataset.

The Convolutional Neural Networks (CNN) are deep networks designed for dealing with the 2-D structure of images (LeCun et al. 1989) (LeCun et al. 1998). The basis of the transform layers are convolution filters that are required to be learned together with the classifier parameters in supervised manner. Once trained, the feature learning stages would be able to produce features of the input samples with the upper layers produces high-level features.

CNNs have large application in learning feature representations for handwriting recognition. The earlier generations of CNN's were evaluated on handwritten digits encountered in postal zip codes (LeCun et al. 1989) (LeCun et al. 1990). Later, several variations were applied in recognizing isolated characters and words/subwords for different scripts including Arabic (AbdulKader 2008), Latin (Simard et al. 2003) (Chellapilla, Shilman, et al. 2006) (Chellapilla, Puri, et al. 2006) (Cireşan et al. 2011) (Liu et al. 2013) (Pan et al. 2015) (Elleuch et al. 2016), Chinese (Chellapilla & Simard 2006) and others (Soman et al. 2013) (Kim & Xie 2015) (Anil et al. 2015).

The cursive nature of the handwritten text makes the application of the application of CNNs to unconstrained handwritten text a challenge. To cope with the problem, either over-segmentation heuristics, i.e., the evaluation of all possible segmentations, or HMMs that can achieve implicit segmentation are integrated with the CNNs to generate segmented characters. In (LeCun et al. 1997), the CNN was applied to read courtesy amount from bank checks. The network was trained using isolated characters. Both the over-segmentation heuristics and implicit HMM-based segmentation were applied on the text extracted from the courtesy amount filed. In (Bluche et al. 2013a) the CNN was applied to extract features for HMM-based handwritten text recognition. The cursive text is explicitly segmented using heuristics or implicitly using HMMs that were trained by handcrafted features. The text segments are delivered to the CNN for feature extraction and recognition. The work of Liu et al. (Liu et al. 2013) integrated CNN and Conditional Random Field (CRF) for recognizing Latin handwritten words. The system assumes that the handwritten words are explicitly segmented into characters, so the CNN was trained on the segmented characters.

Recurrent Neural Networks (RNNs) are another type of supervised DNNs that have large applications in handwriting recognition (Graves et al. 2009). Their ability to reach long context and perform sequence labeling make them good candidate for recognition of cursive handwritten text. This is achieved by the Long Short-Term Memory (LSTM) hidden stages and Connectionist Temporal Classification (CTC) output layer. The Multi-dimensional LSTM (MDLSTM) enables RNNs to learn feature representations for handwritten text starting from the raw pixel intensities. The multi-stage RNNs comprising several MDLSTM hidden layers and CTC output layer was successfully applied to the recognition of Arabic and Latin handwritten text (Graves & Schmidhuber 2009) (Graves 2012) (Liwicki et al. 2012) (Pham et al. 2014) (Bluche, Louradour, et al. 2014) (Maalej et al. 2016) (Maalej & Kherallah 2016) and to machine-printed text (Breuel et al. 2013).

Though the RNNs achieve state-of-the-art accuracies for several scripts, it is unclear whether the performance is due to the discriminative features that have been learned or to the network ability to reach long context and perform accurate segmentation and classification. The two comparative studies (Chherawala et al. 2013) (Bluche, Ney, et al. 2014) experimented with RNN-based handwriting recognition systems using the two classes of features, viz., the features produced by the RNNs and the traditional statistical handcrafted features. Both studies concluded that the handcrafted features outperformed the learned features. Several RNN-based handwritten text recognition systems relied on traditional statistical handcrafted features instead of that produced by the RNN (Frinken & Bunke 2010) (Liwicki et al. 2012).

Deep Believe Networks (DBN) are deep architecture that are trained stage-wise in unsupervised manner (Hinton et al. 2006). The network comprises several stages of generative models (e.g., Restricted Boltzmann Machine (RBM) and auto-encoder) that can be trained layer-wise such that the output of the lower stage is consumed as input for the upper stage. The network that has been trained in unsupervised manner might be fine-tuned in supervised manner using labeled samples in order to adapt the whole network to the target dataset (Salakhutdinov & Hinton 2007). The earlier works of the DBNs was evaluated in the recognition of isolated digits (Hinton et al. 2006) (Salakhutdinov & Hinton 2007). Later, several architectures were proposed for learning representation for handwritten characters and words/sub-words (Porwal et al. 2012) (Elleuch et al. 2015a) (Elleuch et al. 2015b). The DBNs were also applied to learn feature representation for HMM-based recognition systems for cursive handwritten text (Hammerla et al. 2010).

Though deep neural networks achieved state-of-the-art performance in isolated digits and words, two main shortcoming limit their usage in unconstrained open-vocabulary handwritten recognition. The first is the computational complexity of their training procedures (Bengio 2009). The notable performance were achieved by using huge computational power (Krizhevsky et al. 2012) (Cireşan, U. Meier, et al. 2012) (Szegedy et al. 2015) that couldn't be always available. The second is that they require abundant training samples (see Section 2.1.2). Further, most of the supervised Deep Neural Networks (e.g., CNNs) require pre-segmented character samples for training which is difficult to acquire.

2.3. Bag-of-Features Framework

The Bag-of-Features (BoF) framework produces a statistical representation of an image based on the frequencies of occurrences of its local features (O'Hara & Draper 2011). It is inspired by Bag-of-Words (BoW) representation widely used in text retrieval systems. In these systems, a text document is disassembled into individual words. These words are reverted to their stem terms that are defined by a dictionary, e.g., 'does', 'done' and 'doing' all are reverted to their dictionary stem term 'do'. Words that are frequently occurring in text documents like 'a', 'an' or 'the' are ignored from the document and the dictionary, since they do not provide useful discriminative features among documents. Then, the histogram of frequencies of each stem term in the document is computed. Finally, the document is represented by a vector of elements corresponding to terms in the dictionary and its value is the average of occurrences of the terms in the document. This vector is the Bag-of-Words representation of the document. It is called 'Bag' because the order of words in the document is not preserved. This representation is used for indexing text documents in databases and for retrieving documents relevant to user queries. The Bag-of-Features represents images based on local image features. Since Bag-of-Words representation relies on the presence of predefined vocabulary -the set of stem terms-, Bag-of-Features representation needs to define such vocabulary. Once defined, local features extracted from an image are assigned to the appropriate term in the vocabulary and the global histogram is constructed.

The general architecture of the BoF framework is shown in Figure 2.3. The BoF is a two-stage feature learning framework. The first stage is implemented by local descriptors while the second is an instance of unsupervised feature learning.

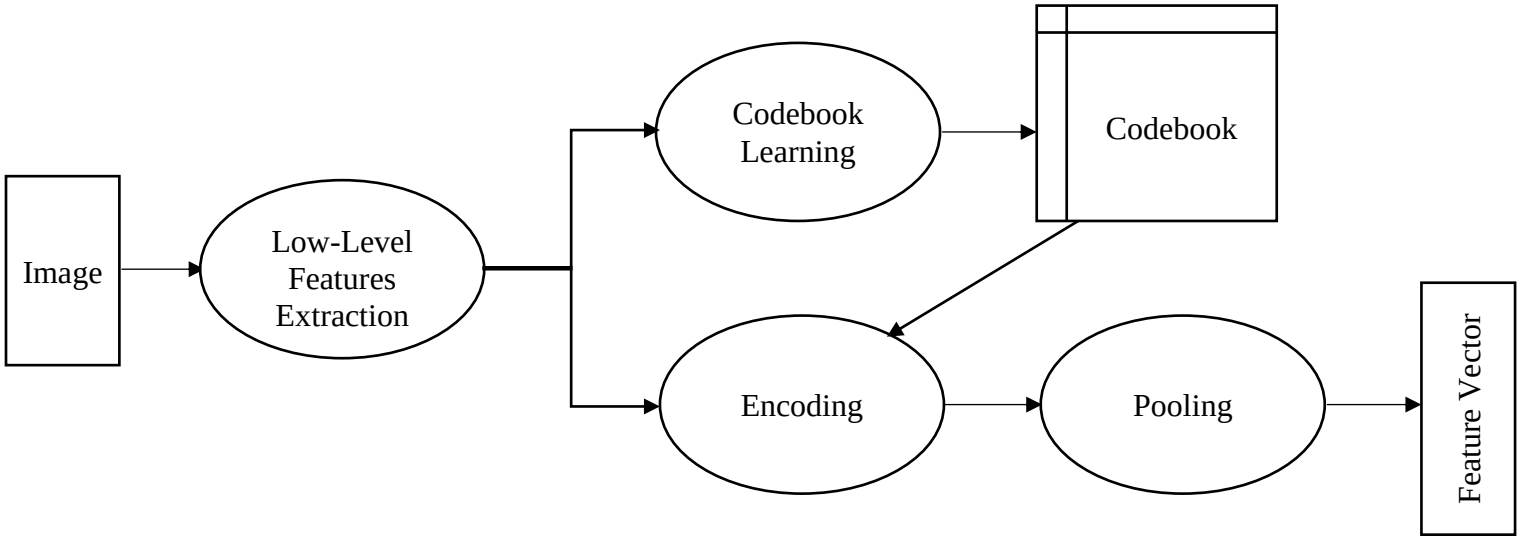


Figure 2.3: The general structure of the BoF Framework

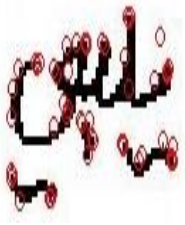
2.3.1. Local Feature Extraction Stage

The earlier versions of the BoF framework (Csurka et al. 2004) (Sivic & Zisserman 2003) relied on the convention of computer vision communities in defining local image features where the extraction of local image features involves two steps: detection and description (Szeliski 2011). The detection locates discriminative regions that can be recognized in the image reliably in presence of illumination changes, occlusion, image translation, rotation, scaling, and affine transformations. The description step encodes the attributes of the detected location and its neighbor pixels into a distinctive and robust representation. Different detection techniques were proposed in computer vision, including Hessian (Beaudet 1978), Harris (Harris & Stephens 1988), Hessian-Laplacian and Harris-Laplacian (Mikolajczyk & Schmid 2001), Difference-of-Gaussian (Lowe 2004) among others. An extensive survey and evaluation of the interest region detectors can be found in (Mikolajczyk & Schmid 2004) (Mikolajczyk et al. 2005) and (Tuytelaars & Mikolajczyk 2007). In addition to interest point detectors, dense and random sampling were used as robust alternatives to interest point detectors (Nowak et al. 2006). Dense sampling provides better coverage of the input image and it was applied in state-of-the-art works in classification and recognition (Law et al. 2014) as well as in text document analysis (Rusiñol et al. 2011) (Rothacker, Rusiñol, et al. 2013) (Tencer et al. 2013). Further, dense sampling is compatible to the receptive field strategies applied in the feature learning frameworks that were originated in machine learning communities (LeCun 2012). Figure 2.4 visualizes interest points detected by common detectors as well as by dense sampling of a handwritten word image.

The second step in defining local image features is to describe the region surrounding the detected regions –or the selected patch in the case of dense sampling- so that the discriminative properties of the region such as intensity, scale, orientation and affine are encoded in a vectoral representation. Several approaches were proposed for interest region description based on pixel intensities (Jurie & Triggs 2005), edge shapes (Belongie et al. 2002), gradient information (Lowe 2004), pixel intensity order (Gupta et al. 2010). An extensive survey and evaluation of the local descriptors can be found in (Mikolajczyk & Schmid 2005) and (Hu et al. 2015).

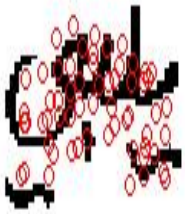
The comparative study of Mikolajczyk & Schmid (Mikolajczyk & Schmid 2005) showed that the descriptors with structure similar to simple-complex cell structure (e.g., SIFT) outperformed others. The state-of-the-art in several BoF comparative studies were achieved by SIFT descriptor and its variations (Law et al. 2014).

(a) Original image



(b) Harris-Laplace

(c) SIFT



(d) SURF

(e) Dense Sampling

Figure 2.4: Interest points detected by common detectors

2.3.2. Unsupervised Feature Learning Stage

The Unsupervised Feature Learning Stage takes the local features extracted in the first stage as input and produces a mid-level representation of the input patterns. Similar to other instances of the Unsupervised Feature Learning, the basis are learned off-line from the training set. BoF relies on k-means clustering for learning the basis, which they are called codewords and they are combined in a codebook. In addition to its simplicity, the Coates and Ng in (Coates & Ng 2012) showed that k-means clustering would learn robust basis when the training samples were decorrelated in advance. Other approaches were proposed in (Jurie & Triggs 2005) (Jianxin & Rehg 2009) (Jain & Doermann 2011) (Zagoris et al. 2014).

The naïve BoF framework applied the vector quantization as a non-linear transform in the encoding layer. To alleviate the side effect of quantization distortion associated with the vector quantization, several works applied more sophisticated non-linear transforms, e.g., the sparse encoding (Boureau, Bach, et al. 2010) (Wang et al. 2010) and soft assignment encoding (van Gemert et al. 2010) (Liu et al. 2011).

In addition to the sum and average pooling that were applied in the early versions of BoF framework, the max pooling which is common in multi-stage frameworks, e.g., HMAX and CNNs was also utilized in the BoF. The analytical and empirical studies of the use of max pooling with the BoF framework showed that max pooling would achieve better performance with the presence of heterogeneous clutter distortion (Boureau, Bach, et al. 2010) (Boureau, Ponce, et al. 2010). The vectoral pooling approach in which the assignment weights of each codeword are pooled at different levels was also proposed in (Avila et al. 2013).

2.3.3. Observing the Spatial Localization

One of the shortcoming of the BoF representation is the lack of spatial localization in the representation, as the representation is a global histogram of the occurrences of the local features. Though several approaches were proposed to encapsulate the spatial information (Lazebnik et al. 2006) (Koniusz & Mikolajczyk 2011) (López-Monroy et al. 2016), the spatial Pyramid Matching (SPM) (Lazebnik et al. 2006) is the common approach. In SPM, the global BoF representation is constructed in consecutive steps. In the first step, the usual histogram is computed. Then the *x*-*y* special area is repeatedly partitioned into sub-regions where an independent BoF histogram is computed of each sub-region. The global BoF representation is the concatenation of all histograms. SPM was frequently applied in document image analysis applications (See Table 2.1).

2.3.4. BoF Framework Improvements

The BoF framework was first proposed for extracting robust representations for image retrieval (Sivic & Zisserman 2003). Since that date, several proposals have been devised to improve the framework performance and to enhance the quality of the representation. The dense sampling strategy (Nowak et al. 2006), the extraction of multi-scale descriptors (Jurie & Triggs 2005) (Bosch et al. 2007), the utilization of sophisticated learning approaches for codebook learning instead of the k-means clustering (Jurie & Triggs 2005) (Jianxin & Rehg 2009) (Jianxin & Rehg 2009) (Jain & Doermann 2011) (Zagoris et al. 2014), the use of sparse and soft encoding schemes (Boureau, Bach, et al. 2010) (Wang et al. 2010) (van Gemert et al. 2010) (Liu et al. 2011), the use of max and vectoral pooling approaches (Boureau, Bach, et al. 2010) (Wang et al. 2010) (Boureau, Ponce, et al. 2010) (Avila et al. 2013), the imposing of the spatial information in the representation (Lazebnik

et al. 2006) (Koniusz & Mikolajczyk 2011) (López-Monroy et al. 2016) and the normalization and transformation of the final representation (Law et al. 2014) (Ionescu & Popescu 2015) are examples of such improvements.

The other tendency for improving the framework is to adapt it to the problem domain and augment the representation by the domain's unique specificities. In (Bailly et al. 2015), the BoF framework was adapted for 1-D time series classification by adjusting the local feature extraction stage to the nature of the single dimensional information. The algorithms for building the Difference-of-Gaussian (DoG) pyramid and detecting the local maxima were modified to fit to the nature of the 1-D time series. The gradient histograms were computed for the gradient sign, as the 1-D gradient don't have magnitude and orientations. In (Ionescu et al. 2013) the framework was adapted for facial expressions recognition. The face image was represented by the presence/absence of the codewords (i.e., the binary BoF representation) as the presence of the codeword is more important than its frequency in recognizing facial expressions. This dissertation is in this stream. Our goal is to adapt the framework to the nature of Arabic handwritten text and augment it by context and the characteristics of the text images and handwritten text.

2.4. BoF Framework for Document Image Analysis Applications

The Bag-of-Features framework was utilized for learning feature representation for handwriting recognition and other related applications known as document image analysis applications, e.g. word spotting, word image query and writer identification and verification. Table 2.1 shows the main specifications of the BoF frameworks that were applied to handwriting recognition and related fields. The work of Rothacker et al.

(Rothacker et al. 2012) is very close to our work. In their work, they proposed BoF as feature learning approach for handwriting recognition based on Hidden Markov Model (HMM). Harris detector was used for detecting the interest point of the text image and SIFT was used for describing the region surrounding them. Codebook sizes between 1500 and 2500 codewords were generated using McQueen K-means with Euclidean distance. SIFT descriptors were first decorrelated using Principle Component Analysis (PCA) before clustering. For quantization, a Gaussian Mixture Model (GMM) was estimated based on the clustering result. Descriptors were assigned to the closest codewords (hard assignment) or to N-closest codewords (soft assignment) based on the estimated GMM. The spatial information was observed by the sliding window strategy that was used by HMM-based systems. The proposed system was evaluated against IfN/ENIT Arabic handwritten text database.

In (Al-dmour & Abuhelaleh 2016) the BoF framework was applied for learning feature representation for holistic handwriting recognition applied to Arabic words. SURF detector and descriptor is applied for extracting local features for word images. The codebook was learned by applying k-means clustering. The vector quantization encoding was used for encoding and the naïve BoF was constructed as a global representation of the image. Support Vector Machine was employed as the classifier. The system was evaluated on the most frequent 18 classes of IfN/ENIT database.

In the body of literature surveyed in this work we found only these two works that applied BoF in handwriting recognition. In both works, and in the other works summarized in Table 2.1, the construction of the BoF representation was identical to what was used for visual

objects and scene images. The document images were handled as similar as the visual objects and scene images. We believe that the adaptation of the framework to the nature of the recognition systems and augmenting it by the characteristics of the Arabic handwritten text would significantly improve the quality of the feature representation and consequently the recognition system performance.

Table 2.2: Summary of BoF frameworks applied to handwriting-related applications

Work	Work Description	Detector	Descriptor	Clustering	Codebook Size	Encoding	BoF Vector
(Al-dmour & Abuhelaleh 2016)	Handwriting recognition	SURF	SURF	K-means	-	Hard Assignment	Naïve BoF
(Rothacker 2011) (Rothacker et al. 2012)	Handwriting recognition	Harris	SIFT	K-means	1500-2500	GMM Hard & Soft Assignments	SPM
(Rusiñol et al. 2015) (Rusiñol et al. 2011)	Word spotting	Dense sampling	SITF	K-means	1500 -32768	NN Hard Assignment	SPM
(Aldavert et al. 2015)	Word spotting	Dense sampling	HoG	K-means	32-16384	Locality-constrained Linear Coding	SPM
(Aldavert et al. 2013)	Word spotting	Dense sampling	SIFT	K-means	4096	3-NN Soft Assignment	SPM
(Shekhar & Jawahar 2012)	Word image retrieval	Harris	SIFT	Hierarchical K-Means	10000	Hierarchical K-Means Hard Assignment	SPM
(Shekhar & Jawahar 2013)	Word spotting	Harris	SIFT	Hierarchical K-Means	-	Sparse code and locality constrained linear coding	SPM
(Rothacker & Fink 2015) (Fink et al. 2014) (Rothacker, Rusiñol, et al. 2013) (Rothacker, Fink, et al. 2013)	Word spotting	Dense sampling	SIFT	K-means	1024 - 4096	Hard Assignment	Naïve BoF - SPM
(Sudholt et al. 2015)	Word spotting	Dense sampling	Projected SIFT	K-means	1024	Hard Assignment	SPM

(En et al. 2016)	Pattern Spotting	Dense sampling	SIFT	K-means GMM	500-10000	Hard Assignment	Naïve BoF VLAD FV
(Christlein et al. 2015)	Writer Identification	Dense sampling	Contour-Zernike Moments	K-means	100	Hard Assignment	VLAD
(Fiel & Sablatnig 2013)	Writer Identification and Writer Retrieval	SIFT	SIFT	GMMs	50	cosine distance with α-normalization	Naïve BoF
(Fiel & Sablatnig 2012)	Writer Identification and Writer Retrieval	SIFT	SIFT	K-means	300	NN Hard Assignment	Naïve BoF
(Jain & Doermann 2011)	Writer Identification and Writer Retrieval	K-Adjacent Segments	KAS features	Affinity propagation	300	KNN	Naïve BoF
(Gandhi & Jawahar 2013)	Detecting Cut-and-Past in document images	SIFT	SIFT	K-means	20000	Mixture of Homographies Model	Naïve BoF
(Tencer et al. 2013)	Sketch-based image retrieval system	Random sampling Dense Sampling	Discrete Distance Transformation	K-means	500-1000	NN	Naïve BoF
(Zagoris et al. 2014)	Handwritten vs. Machine-printed text. discrimination	SIFT	SIFT	SGONG network	Determined by SGONG	NN Hard Assignment	Naïve BoF

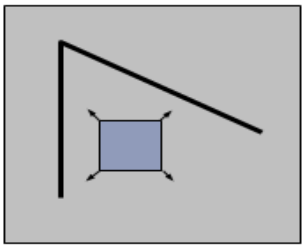
2.5. Algorithms for Implementing BoF Framework

In this section, we present a brief background for the algorithms that we are used in this work in implementing the BoF framework.

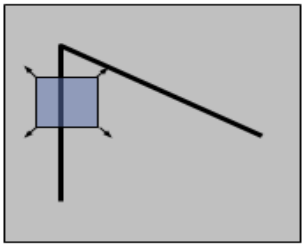
2.5.1. Local Feature Detection

The local feature detection is the first step in the Local Feature Extraction Stage of the BoF framework (Section 2.3.1). The aim of this step is to locate discriminative regions that can be recognized in the image reliably in presence of illumination changes, occlusion, image translation, rotation, scaling, and affine transformations. In this work, we evaluate the interest point detectors and dense sampling.

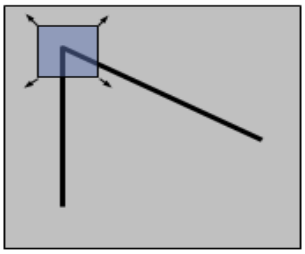
Interest point detectors are computer vision techniques developed mainly for image alignment applications and are widely used in image matching applications. These operators are based on the observation that "*distinctive regions can be located on points that show illumination changes in two directions"*. This can be observed by looking through a small window around image pixels as shown Figure 2.5. For a point lying in a uniform region (a), shifting the window in any direction doesn't show any change in the intensity. Similarly, for a point on an edge (b), we can only notice changes perpendicular to the edge but not along the edge direction. However, for an intersection point (a corner) (c), shifting the window in any direction gives a large change in intensity (Grauman & Leibe 2011).



(a) Uniform region: no change in all directions



(b) On edge: changes perpendicular to the edge but not along the edge direction.



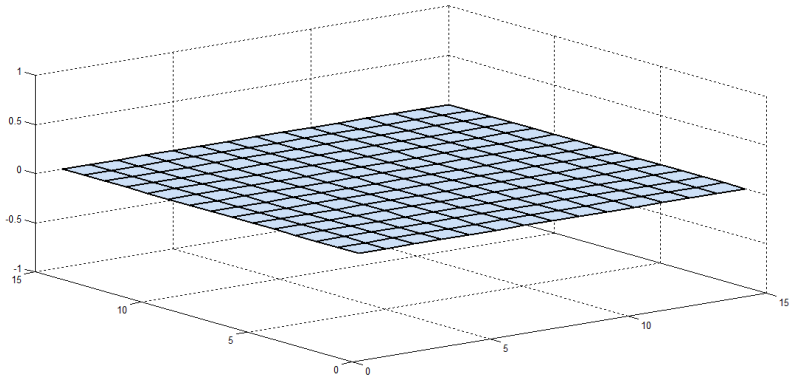
(c) On corner: large change in all direction

Figure 2.5: Effects of shifting small window around different image regions

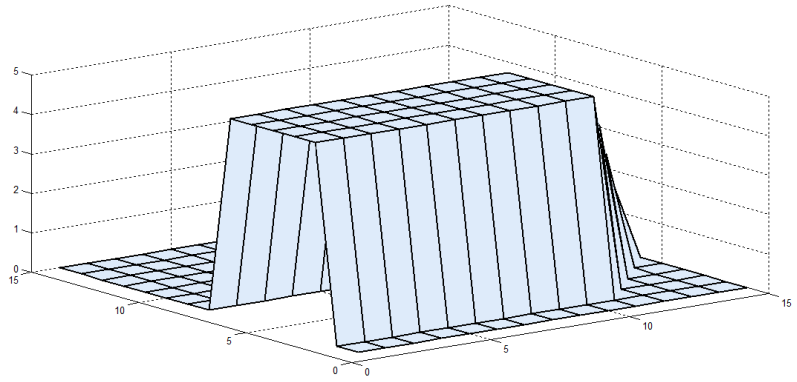
Moravec in (Moravec 1980) analyzed the intensity change based on the *local autocorrelation* that can be measured by the *Summed Square Difference* (SSD), E(u,v):

$$E(u,v) = \sum_{x,y} w(x,y).\left[I(x+u\,,y+v) - I(x\,,y)\right]^2 \qquad (1)$$

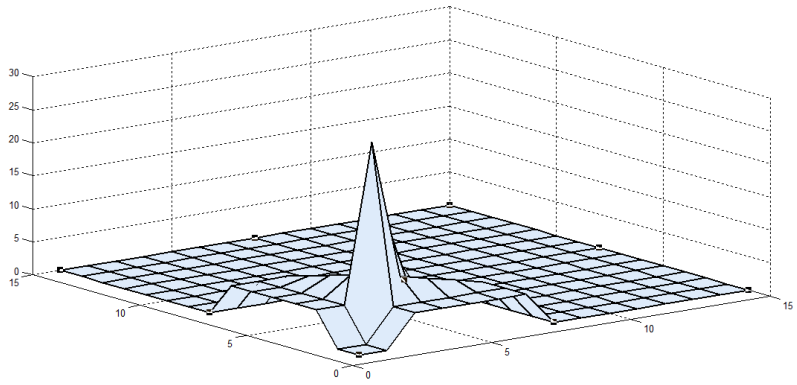
where *I* is the image intensity, *(x,y)* iterates over all pixels in the image patch, *(u,v)* indicates the shift in x and y directions and *w* is a two dimensional window used to clip the currently processed patch of the image. *w* has a value of 1 within the patch and 0 outside. *E(u,v)* is a measure of the similarity between intensities of patches' pixels when the patch is shifted in *(u,v)* direction. Small value of *E(u,v)* indicates that the intensities of patches' pixels are similar in *(u,v)* direction, while large value indicates that the intensities of patches' pixels are dissimilar in *(u,v)* direction. Evaluating *E* for all possible values of *(u,v)* over image patches determined by the window in the three images of Figure 2.5 exhibits different behavior as shown in Figure 2.6. In the flat region, E is a flat surface (a) indicating no changes in intensity when the window is shifted. *E* over the edge region (b) has maximum on multiple points indicating the changes perpendicular to the edge direction. In (c) *E* has maximum in one point indicating a corner. Moravec used *E* as the measure for detecting corner-like structures in images by evaluating *E* for each image pixel and searching for points that shows local maximum with respect to their neighbors. Then it returns points whose value is above a predefined threshold.



(a) *E* over the flat region



(b) *E* over the edge



(c) *E* over the corner

Figure 2.6: Evaluation Summed Square Difference E(u,v) over different image patches

2.5.1.1. Harris Detector

Harris and Stephens developed an interest point detector that responds to corner-like images structures based on Moravec detector (Harris & Stephens 1988). In their work, they expand Equation (1) using Taylor series, considering only the first derivative terms as an approximation. They also replace the two dimensional box window function *w(x,y)* by a two dimensional Gaussian window $G(x,y) = e^{-\frac{(x^2+y^2)}{2\sigma^2}}$ to reduce the noise effects:

$$
\begin{aligned}
E_{u,v} &\approx \sum_{x,y} G(x,y).\left[I(x\,,y) + uI_x + vI_y - I(x\,,y)\right]^2 \\
&= \sum_{x,y} G(x,y).\left[u^2 I_x^2 + 2\,uvI_xI_y + v^2 I_y^2\right] \\
&= \begin{bmatrix} u & v \end{bmatrix} \left(\sum_{x,y} G(x,y). \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix} \\
&= \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}, \qquad M = \sum_{x,y} G(x,y). \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}
\end{aligned}
$$

Instead of looking for local maxima of E, they analyze the eigenvalues of the auto-correlation matrix M (sometimes called second moment matrix) and find that a point with two large eigenvalues indicates a corner. Further, they design a corner response measure R based on the determinant and trace of M:

$$R = det(M) - k.\left(trace(M)\right)^2$$

Where k is a constant whose value is determined experimentally to be in the interval [0.04 , 0.06]. A point is a corner if the value of R is greater than a predefined threshold value. Steps involved in Harris detector are shown in ALGORITHM 2.1 (Szeliski 2011).

ALGORITHM 2.1 : Harris Detector

1. Compute first derivatives I_x , I_y at each pixel using Gaussian kernel.
2. Compute the auto-correlation matrix M in a Gaussian window around each pixel.
3. Compute corner response function R.
4. Threshold the result of (3).
5. Find local maxima of the response function.

Harris detector is invariant to noise (due to the smoothing step by Gaussian kernels), illumination changes (due to adapting threshold value) and rotation (rotation in image view only rotates the direction of eigenvectors but doesn't change their magnitude). However, Harris detector is not invariant to scale. In this work, we use a fixed-size patch of 24 pixels centered at each detected region. We found experimentally that this size is the best among the four sizes that are used in the dense sampling approach as discussed below.

2.5.1.2. Harris-Laplace Detector

The Harris-Laplace detector is the scale-invariant version of Harris (Mikolajczyk & Schmid 2001). It was build based on the study of Lindeberg (Lindeberg 1993) that showed that the automatic detection of interest points invariance to image scale can be achieved by convolving the image with a Gaussian (or a Gaussian derivative) kernel of different sizes and searching the resulting 3-D structure (space + scale) for maxima (Szeliski 2011). Harris-Laplacian involves two phases. In the first phase, a space-scale representation is built by convolving the image with Gaussian functions differing in σ values. Then, the

Harris detector is applied to detect the interest points at each level. In the second phase, another space-scale representation is built by convolving the image with Laplacian functions having the σ values used in the first space-scale representation. The interest points are those that remain maximal in the Laplacian scale corresponding to the scale at which they were detected.

2.5.1.3. Dense Sampling

In the dense sampling approach, the image spatial area is divided into a grid of overlapping fixed-size patches. We use four grids of different patch sizes (viz. 16, 24, 32 and 40 pixels) and a stride of 8 pixels in the four grids. The preliminary experiments showed that multi-size sampling has better performance than single-size. These four grids were used in other works that employed dense sampling, including (Law et al. 2014) and (Chatfield et al. 2011). Though a stride shorter than 8 pixels would produce better accuracy, this value makes a balance between the accuracy and the computational performance especially when the GMM is applied for encoding (Section 5.5.5).

Representative Regions Representation

Each representative region *i* (either the one reported by the detector or the dense samples) is represented by a 3-D vector $\mathbf{l}_i = [x, y, s]^{\mathrm{T}}$ comprises the spatial (x , y)-coordinates and the spatial size (in pixels) of the representative region. The representative regions vectors of the image are concatenated column-wise into a matrix $\mathbf{L} \in \mathbb{R}^{3\times N}$ where N is the number representative regions in the image.

2.5.2. Local Feature Description

The Local Feature Description is the second step in the Local Feature Extraction Stage of the BoF framework (Section 2.3.1). In this work, we apply SIFT descriptors (Lowe 2004) to describe the representative regions. SIFT uses *local image gradients* relative to interest point scale and orientation for describing the interest point region. The scale is used to select the proper image from the image pyramid and the orientation is used to rotate the interest point region to the *standard orientation*, the one in which the dominant orientation is upward. SIFT descriptor is constructed by computing the local gradient of each pixel in the region. Once computed, the region is divided into 4×4 sub-regions. Within each sub-region, the gradient magnitudes of the pixels are weighted by a Gaussian kernel and accumulated into 8-bin orientation histogram, giving a 128-D vector. This vector is further normalized to unity and used as a representation to the representative region.

Given the representative regions matrix $\mathbf{L}$, SIFT produces a 128-D descriptor $\mathbf{x}_i \in \mathbb{R}^{128}$ for each region $\mathbf{l}_i$. The image descriptors are concatenated column-wise into a matrix $\mathbf{X} \in \mathbb{R}^{128\times N}$.

2.5.3. Low-Level Features De-correlation

The SIFT descriptors are de-correlated before applying the learning algorithm and the encoding function. De-correlation acts as the preprocessing phase for the second stage (the unsupervised feature learning stage). The study of Coates and Ng (Coates & Ng 2012) showed that linear de-correlation has significant influence on the performance of the k-means clustering. We use the Principle Component Analysis (PCA) whitening where each

descriptor vector $\mathbf{x}$ is linearly transformed to a new vector $\tilde{\mathbf{x}}$ such that its components are uncorrelated with unity variances, i.e., $E[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T] = \mathbf{I}$ (Hyvärinen & Oja 2000).

The principle components are learned from the training set by selecting representative one million descriptors ($\mathbf{S} \in \mathbb{R}^{128\times10^6}$) from the training samples. To learn the mean $\boldsymbol{\mu}$ and the principle components $\mathbf{P}$ from the representative descriptors, ALGORITHM 2.2 is applied.

For a descriptor vector $\mathbf{x}_i$, the PCA whiting is applied to produce a new vector $\tilde{\mathbf{x}}_i$ such that:

$$\tilde{\mathbf{x}}_i = \mathbf{P}\times(\mathbf{x}_i - \boldsymbol{\mu})$$

In the following sections, we will use $\mathbf{x}_i$ to denote to the de-correlated descriptors.

ALGORITHM 2.2 : Estimate the mean and the principle components of the training samples

1. Select 1 million descriptors from the training samples $\mathbf{S} \in \mathbb{R}^{128\times10^6}$:
    a. Select 30% random images from the training set of each class.
    b. Select 50% random descriptors from each image.
    c. Shuffle the set and select one million descriptors.
2. Compute the mean $\boldsymbol{\mu}$ and the covariance $\boldsymbol{\Sigma}$ of $\mathbf{S}$.
3. Compute the diagonal matrix of eigenvalues $\mathbf{U}$ and the eigenvectors matrix $\mathbf{V}$ of $\boldsymbol{\Sigma}$, such that $\boldsymbol{\Sigma} = \mathbf{VUV}^{\mathrm{T}}$.
4. Sort the columns of $\mathbf{U}$ and $\mathbf{V}$ in descending order according to the computed eigenvalues.
5. Denote $\mathbf{U}^{-1/2}$ to the reciprocal of the square-roots of $\mathbf{U}$ (element-wise), compute the principle component matrix $\mathbf{P}$:

$$\mathbf{P} = \mathbf{U}^{-1/2}\times\mathbf{V}^{\mathrm{T}}$$

6. Return the mean $\boldsymbol{\mu} \in \mathbb{R}^{128}$ and the principle components $\mathbf{P} \in \mathbb{R}^{128\times128}$.
7. If dimensionality reduction is required, return only the first D columns of $\mathbf{P}$, i.e., $\mathbf{P} \in \mathbb{R}^{128\times D}$.

2.5.4. Codebook Learning

As in unsupervised feature learning frameworks, the basis are learned offline using unlabeled samples. In BoF literature, the basis are named codewords and they are grouped in a codebook. The codebook is learned by the K-means clustering algorithm (Xu & Wunsch 2005). The simplicity and computational efficiency of the K-means clustering make it the standard algorithm for feature learning framework originated in the field of computer vision. Further, the study of Coates and Ng (Coates & Ng 2012) showed that the K-means clustering could achieve the performance of advanced learning algorithms, e.g., auto-encoders and Restricted Boltzmann Machines (RBMs), with stronger encodings than the vector quantization encoding. Consequently, we build our framework on the K-means and investigate several encodings.

To learn a codebook $\mathbf{C} \in \mathbb{R}^{D\times K}$ of *K* codewords, the K-means clustering is applied on a set of million (de-correlated) descriptors ($\mathbf{S} \in \mathbb{R}^{D\times 10^6}$) selected randomly from the training samples. The K-means clustering iteratively computes a set of *K centroids:*

$$\mathbf{C} = \{\mathbf{c}_i \,|\mathbf{c}_i \in \mathbb{R}^D, i = 1, 2, \ldots\ldots K\}$$

and *vector-to-centroid assignments*

$$\{a_i \mid a_i \in \{1, 2, \ldots\ldots, K\}, i = 1, 2, \ldots. 10^6\}$$

such that the assignment error

$$\sum_{i=1}^{N} \left\|\mathbf{x}_i - \mathbf{c}_{a_i}\right\|^2$$

is minimal.

The learned codebook is represented as a matrix $\mathbf{C} \in \mathbb{R}^{D \times K}$ where each column is a visual codeword.

2.5.5. Encoding

The encoding layer utilizes the learned codebook **C** to transform each low-level feature vector $\mathbf{x}_i \in \mathbb{R}^D$ into a vector $\mathbf{u}_i \in \mathbb{R}^K$:

$$\mathbf{u}_i = \left[u_{i,1}, u_{i,2}, \ldots\ldots, u_{i,K}\right]^T$$

such that

$$u_{i,j} \in [0, 1]$$

and

$$\sum_{j=1}^{K} u_{i,j} = 1$$

The component $u_{i,j}$ is the contribution of the codeword $\mathbf{c}_j$ in representing $\mathbf{x}_i$. The value of $u_{i,j}$ depends on the encoding scheme. For an image sample of N descriptors represented column-wise as $\mathbf{X} \in \mathbb{R}^{\mathrm{D} \times \mathrm{N}}$, the encoding layer produces a matrix $\mathbf{U} \in \mathbb{R}^{\mathrm{K} \times \mathrm{N}}$. The i^{th} column of **U**, $\mathbf{u}_i$, is the encode of the local feature $\mathbf{x}_i$. The encoding schemes we evaluated in this work are presented in the following subsections.

2.5.5.1. Hard Assignment Encoding

In hard-assignment, a local feature is assigned to a codeword based on the minimum Euclidean distance, i.e.,

$$u_{i,j} = \begin{cases} 1 & if\ \ j = \arg\min_{j=1,2,\ldots,K} d(\mathbf{x}_i\ , \boldsymbol{c}_j) \\ 0 & otherwise \end{cases}$$

2.5.5.2. K-Nearest-Neighbor Assignment Encoding

In k-Nearest-Neighbor Assignment coding, the local feature $\mathbf{x}_i$ is assigned to the nearest k codewords $\mathcal{N}_k(\mathbf{x}_i)$ based on the similarity $sim(\mathbf{x}_i\ , \mathbf{c}_j)$ defined as:

$$sim(\mathbf{x}_i\ , \mathbf{c}_j) = \begin{cases} \dfrac{1}{1 + d(\mathbf{x}_i\ , \mathbf{c}_j)} & if\ \boldsymbol{c}_j \in \mathcal{N}_k(\mathbf{x}_i) \\ \\ 0 & otherwise \end{cases}$$

Therefore,

$$u_{i,j} = \frac{sim(\mathbf{x}_i\ , \boldsymbol{c}_j)}{\sum_{k=1}^{K} sim(\mathbf{x}_i\ , \boldsymbol{c}_k)}$$

2.5.5.3. Localized Soft-Assignment Encoding

In Localized Soft-Assignment Encoding (Liu et al. 2011) , the local feature $\mathbf{x}_i$ is assigned to the nearest k codewords $\mathcal{N}_k(\mathbf{x}_i)$ based on the code uncertainty $CU(\mathbf{x}_i\ , \mathbf{c}_j)$ defined as:

$$CU(\mathbf{x}_i\ , \mathbf{c}_j) = \begin{cases} e^{-\beta d(\mathbf{x}_i\ , \mathbf{c}_j)} & if\ \mathbf{c}_j \in \mathcal{N}_k(\mathbf{x}_i) \\ \\ 0 & otherwise \end{cases}$$

Therefore,

$$u_{i,j} = \frac{CU(\mathbf{x}_i, \mathbf{c}_j)}{\sum_{k=1}^{K} CU(\mathbf{x}_i, \mathbf{c}_k)}$$

The value of β is tuned in the cross-validation.

2.5.5.4. Soft-Assignment Encoding

In the soft-assignment encoding, a local feature is assigned to all codewords. Gaussian Mixture Models (GMMs) of *K* components are estimated from the set of descriptors *S* used in codebook learning. The initial value of the mixture means is the codebook $\mathbf{C}$. The GMMs is represented by $G = (\boldsymbol{\mu}; \boldsymbol{\Sigma}; \mathbf{w})$ for the means, covariances and weights of the mixture components. Once estimated, the j^{th} component of $\mathbf{u}_i$, $u_{i,j}$, is given by:

$$u_{i,j} = \mathbf{w}_j \times \frac{1}{(2\pi)^{\frac{K}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_j)}$$

The value of $u_{i,j}$ is the posterior probability of the descriptor $\mathbf{x}_i$ belonging to the component *j*. To ensure that the components are summed up to 1, the vector $\mathbf{u}_i$ is normalized to unity:

$$u_{i,j} = \frac{u_{i,j}}{\sum_{k=1}^{K} u_{i,k}}$$

2.5.6. Pooling

Pooling operation aggregates the image's encoding vectors $\mathbf{U} \in \mathbb{R}^{K \times N}$ into a single vector $\in \mathbb{R}^K$:

$$\mathbf{z} = [z_1, z_2, \ldots\ldots, z_K]^T$$

The value of z_i depends on the pooling operation. Two pooling operations are used in this work, viz., average pooling and max pooling.

2.5.6.1. Average Pooling

In average pooling, the i^{th} component of **z** is given by:

$$z_i = \frac{1}{N}\sum_{j=1}^{N} u_{j,i}$$

2.5.6.2. Max Pooling

In max pooling, the i^{th} component of **z** is given by:

$$z_i = \max_{j=1,2,\ldots N} u_{j,i}$$

2.5.7. The BoF Representation

The final image representation is obtained by normalizing **z** to unity:

$$z_i = \frac{z_i}{\sum_{k=1}^{K} z_k}$$

2.6. Gabor Filters Features

In this work, we utilize Gabor filters in the first stage of the BoF framework. In this section we present a brief background of Gabor filter features.

Gabor filter features achieved recognizable performance in handwriting recognition (Mahmoud 2008) (Mahmoud 2009) (Haboubi et al. 2009) (Chen et al. 2010) (Mahmoud & Al-Khatib 2010) (Porwal et al. 2012) (Elzobi et al. 2012) (Elzobi et al. 2014), machine-printed text recognition (Al-Jamimi & Mahmoud 2010) (Zaafouri et al. 2015), writer identification (Said et al. 2000) (Helli & Moghaddam 2010), script identification (Busch et al. 2005) (Pan et al. 2005) (Rajput & Anita 2011) and handwritten vs. machine-printed text identification (Echi & Saidani 2014). Gabor filters were utilized in implementing the first stage of the early feature learning frameworks inspired by the Hubel and Wiesel model, e.g., the Neocognitron (Fukushima 1980), Cresceptron (Weng et al. 1992) and HMAX (Poggio & Riesenhuber 1999). The modern unsupervised feature learning algorithms that were applied directly on the raw pixel intensities (e.g., auto-encoders and RBM) are eventually ended by learning local filters similar to Gabor filters (Jarrett et al. 2009) (Coates, Lee, et al. 2011) (LeCun 2012). This indicates the power of Gabor filters in emulating the functionality of simple-cells of the primary cortex. Recently, Gabor filters features were utilized in the design of a SIFT-like local descriptor coined "*The Biologically Inspired Local Descriptor* (BILD)" (Zhang et al. 2014).

Gabor filter features are obtained by convolving text images with a Gabor filter bank of different scales and orientations and the convolution amplitude is used as the features.

2.6.1. Gabor Filter Bank

The Gabor filter bank consists of a set of Gabor filters. A 2-D Gabor filter $g(x, y; u, v, \sigma)$ is a band-pass filter of a bandwidth bounding by a Gaussian envelope of σ^2 variance, centered on a carrier frequency (u, v). Though it can be represented mathematically in different formula (Movellan 2008), in this work, the carrier frequency (u, v) is represented in the polar coordinate with a magnitude f and orientation θ :

$$f = \sqrt{u^2 + v^2}$$

$$\theta = tan^{-1}\frac{v}{u}$$

Accordingly, the carrier frequency (u, v) is expressed as:

$$u = f\cos\theta$$

$$v = f\sin\theta$$

To make the filter size relative to the image spatial dimensions, the frequency magnitude f is expressed in terms of its wavelength λ:

$$\lambda = \frac{1}{f}$$

Further, to make the Gaussian envelope shape relative to the filter size, the parameter σ is expressed in terms of the carrier wavelength λ:

$$\sigma = k\lambda \quad , k \text{ is a scalar factor}$$

Therefore, the Gabor filter $g(x, y\,; u, v, \sigma\,)$ is fully defined by two parameters:

1. The carrier wavelength λ in pixels.
2. The carrier orientation θ in degrees.

Accordingly, the 2-D Gabor filter is expressed in polar coordinates as:

$$g(x, y\,;\,\lambda\,, \theta\,) = \frac{1}{2\pi(k\lambda)^2}\, exp\left(-\frac{x'^2 + y'^2}{2(k\lambda)^2}\right) exp\left(2\pi j\left(\frac{x\cos\theta + y\sin\theta}{\lambda}\right)\right) \qquad (1)$$

where $x'\ and\ y'$ indicate that the Gaussian envelope is rotated towards θ (the carrier orientation):

$$x' = \quad x\cos\theta + y\sin\theta \qquad (2)$$

$$y' = -x\sin\theta + y\cos\theta$$

Using the Euler's Identity and observing the value of x' in eq. (2), the complex sinusoid in eq. (1) is rewritten as:

$$exp\left(2\pi j\left(\frac{x\cos\theta + y\sin\theta}{\lambda}\right)\right) = \cos\frac{2\pi x'}{\lambda} + j\sin 2\pi\frac{2\pi x'}{\lambda}$$

The Gabor filter $g(x, y\ ;\ \lambda\ , \theta\ )$ is decomposed into the real part (the even filter) g_e and the imaginary part (the odd filter) g_o:

$$g(x, y\ ;\ \lambda\ , \theta\ ) =\ g_e(x, y\ ;\ \lambda\ , \theta\ ) +\ j\ g_o(x, y\ ;\ \lambda\ , \theta\ )$$

where,

$$g_e(x, y\ ;\ \lambda\ , \theta\ ) =\ \frac{1}{2\pi(k\lambda)^2}\ exp\left(-\frac{x'^2 +\ y'^2}{2(k\lambda)^2}\right) \cos\frac{2\pi x'}{\lambda}$$

and

$$g_o(x, y\ ;\ \lambda\ , \theta\ ) =\ \frac{1}{2\pi(k\lambda)^2}\ exp\left(-\frac{x'^2 +\ y'^2}{2(k\lambda)^2}\right) \sin\frac{2\pi x'}{\lambda}$$

2.6.2. Gabor Filter Response

The Gabor filter response $G(x, y\ ;\ \lambda\ , \theta\ )$ is obtained by convolving the image $I(x, y)$ with the Gabor filter $g(x, y\ ;\ \lambda\ , \theta\ )$:

$$\begin{aligned} G(x, y\ ;\ \lambda\ , \theta\ ) &= I(x, y) *\ g(x, y\ ;\ \lambda\ , \theta\ ) \\ &=\ I(x, y) *\ g_e(x, y\ ;\ \lambda\ , \theta\ ) + j\ I(x, y) *\ g_o(x, y\ ;\ \lambda\ , \theta\ ) \\ &=\ G_e(x, y\ ;\ \lambda\ , \theta\ ) + j\ G_o(x, y\ ;\ \lambda\ , \theta\ ) \end{aligned}$$

The response magnitude $|G(x, y\ ;\ \lambda\ , \theta\ )|$ is defined as:

$$|G(x, y\ ;\ \lambda\ , \theta\ )| =\ \sqrt{G_e^2(x, y\ ;\ \lambda\ , \theta\ ) +\ G_o^2(x, y\ ;\ \lambda\ , \theta\ )}$$

For computational efficiency, the Gabor filter and the image are transformed to the frequency domain using the Fast Fourier Transform ($\mathcal{F}$) where the convolution is performed as multiplication:

$$G(x, y\,;\,\lambda\,,\theta\,) = \mathcal{F}^{-1}\{\mathcal{F}\{I(x,y)\} \times \mathcal{F}\{g(x, y\,;\,\lambda\,,\theta\,)\}\}$$

The image is convolved with a Gabor filter bank of different wavelengths and orientations and the response magnitude is computed. For a Gabor filter bank of M wavelengths Λ and N orientations Θ defined as

$$\Lambda = \{\lambda_m \quad | \; m = 1,2, \ldots \ldots, M\}$$

$$\Theta = \left\{\theta_n = \frac{\pi n}{N} \quad | \; n = 1, 2, \ldots \ldots \ldots, N\right\}$$

The Gabor filter bank $GFB_{\Lambda\,,\Theta}$ is a set of $M \times N$ filters:

$$GFB_{\Lambda\,,\Theta} = \{g(x, y\,;\,\lambda_m\,,\theta_n) \mid \quad \lambda_m \in \Lambda\;, \theta_n \in \Theta\}$$

The image is convolved with each filter and the convolutional magnitude is computed. The convolutional magnitude is represented to the BoF framework as 4-D matrix $\mathbf{R} \in \mathbb{R}^{X \times Y \times N \times M}$:

$$\mathbf{R} = \left[\, a_{x,y,n,m} = \left|G(x, y\,;\,\lambda_m\,,\theta_n\,)\right| \,\right]_{X \times Y \times N \times M}$$

The 4-D **R** representation is visualized in Figure 2.7.

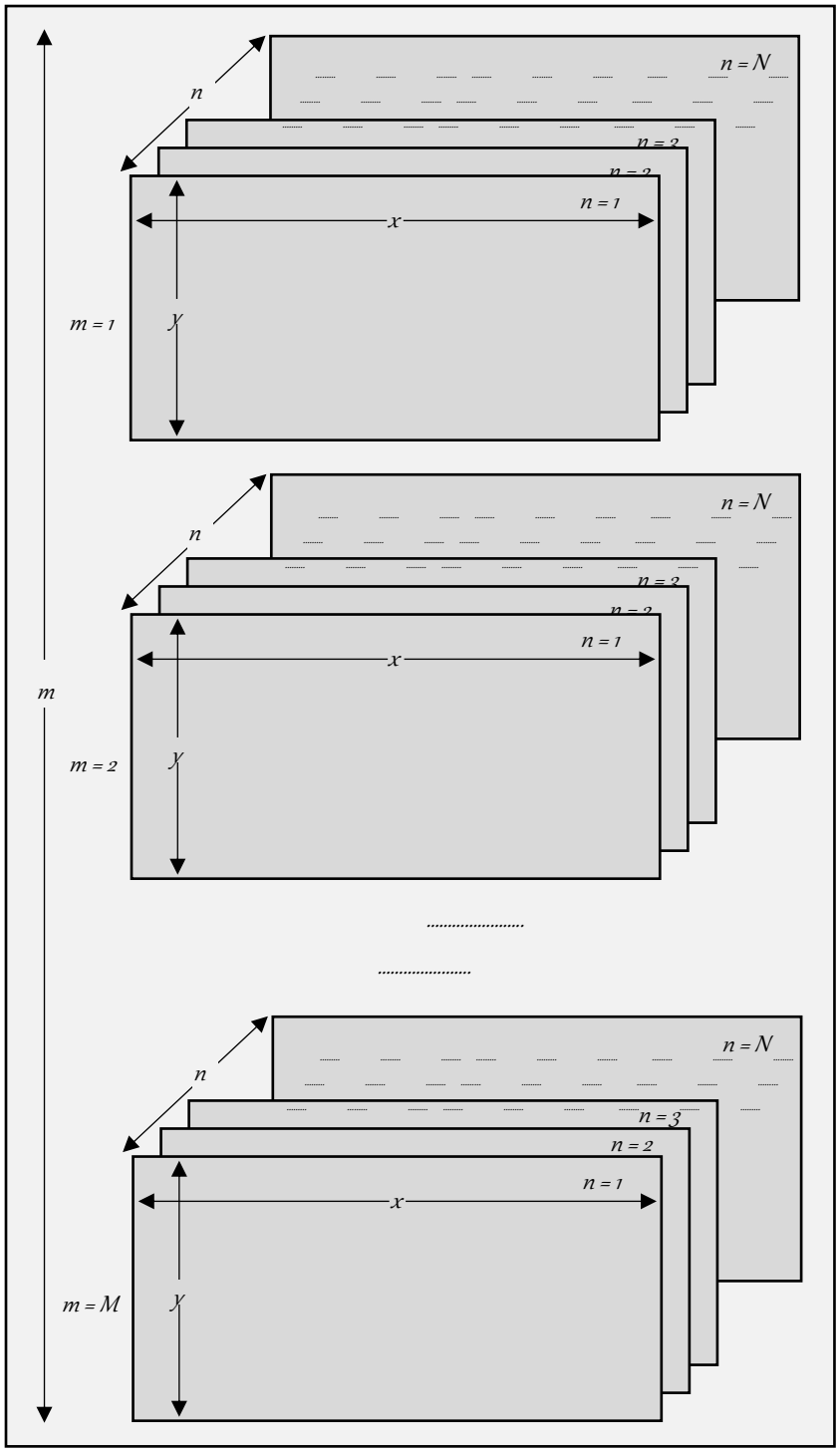


Figure 2.7: The 4-D Matrix represented the Gabor Filter Response of an image

2.7. Conclusions

The feature learning frameworks invented for vision applications have common architecture that enable them to tolerate the different types of variations associated with images. The local descriptor algorithms have similar architecture to that of the feature learning frameworks. This explains the high performance they achieved in several vision applications.

Several feature learning frameworks were applied to handwriting recognition with notable recognition performance, particularly in the recognition of isolated digits and words. Their application to the recognition of cursive handwritten text, however, is limited as most of these frameworks require per-segmented characters for training.

The Recurrent Neural Networks (RNNs) that utilize the Multi-dimensional Long Short-Term Memory (MDLSTM) and Connectionist Temporal Classification (CTC) achieved notable recognition performance in cursive handwritten text of several scripts, including Arabic. However, the studies that compared the performance of the learned features with statistical handcrafted features using RNN-based recognition system showed that the high performance is due to the network ability to reach long context and perform accurate segmentation and classification rather than to the learned features.

The unsupervised feature learning frameworks with self-taught learning seem the appropriate choice for cursive handwritten text recognition systems. This combination could learn robust representations using small patches cropped randomly from handwritten text images. Therefore, the need for per-segmented characters is alleviated. Further, unsupervised feature learning frameworks can be easily adjusted based on the context and

the characteristics of text images and handwritten text. The Bag-of-Features (BoF) framework is an instance of this category that previously achieved prominent performance in image categorization and utilized by several research groups for handwriting recognition, word spotting and writer identification.

The extensive review we carried on the applications of feature learning in the recognition of Arabic handwritten text showed that all of the works were applied to limited vocabulary datasets that lack the naturalness of Arabic handwritten text. To nominate these frameworks for dealing with natural unconstrained Arabic handwritten text, they must be subject to extensive assessment using comprehensive dataset that gives true representation of Arabic handwritten text.

CHAPTER 3

FEATURE LEARNING FRAMEWORK FOR HOLISTIC HANDWRITING RECOGNITION

In this chapter, we present the baseline for BoF framework we will use throughout the dissertation. We integrated the framework baseline with a holistic handwriting recognition system and evaluated it against two Arabic handwritten text datasets, viz., the non-touching Arabic Indian digits and the Arabic sub-words datasets of CENPARMI Bank check database. Since BoF framework involves several steps that can be implemented by a variety of techniques, Section 3.1 provides thorough investigation of several techniques, focusing on the options that have impact on the quality of the produced features. In Section 3.2 we present the approaches we proposed for exploiting the characteristics of text images and handwritten text in enhancing the performance of the BoF framework. In Section 3.3 we present our approaches for utilizing Gabor filter response features in BoF framework. The conclusion remarks are presented in Section 3.4.

3.1. Feature Learning Baseline

The feature learning framework we present is a two-stage unsupervised feature learning framework, specifically, the Bag-of-Features (BoF) framework. The BoF framework has interesting properties that are appropriate for handwriting recognition.

1. The framework can be easily adjusted based on the characteristics of the handwritten text. This is one advantage of the unsupervised feature learning frameworks (See Section 2.1.2)
2. The first stage is implemented using local descriptors that are invariant to image transformations and deformations. Particularly, local descriptors that are based on the visual appearance make the framework script-independent, so it can be adapted to handwriting/machine-printed recognition systems of different scripts.
3. The self-taught learning paradigm enables the use of unlabeled low-level features in learning the basis (codebook elements) of the second stage. This alleviates the need for pre-segmented characters for learning the codebook elements.
4. The learning algorithm is computationally efficient and at the same time has achieved the performance of sophisticated learning approaches (Coates, Lee, et al. 2011).
5. The encoding and pooling layers of the second stage could be implemented by different algorithms without affecting the lower stage.

3.1.1. Feature Learning Baseline Specifications

Since BoF framework involves several stages that can be implemented in diverse approaches, we have implemented different approaches for each stage. Table 3.1 summarizes the main configuration of the BoF framework used in this work. The Harris detector, Harris-Laplace detector and dense sampling were implemented for selecting representative image regions. These three approaches are good representative for the three categories for representative regions selection (Section 2.5.1). Harris detector detects

salient regions at fixed scale. Harris-Laplace detector is scale-invariant and can detect salient regions at different scales. The dense sampling approach is the brute force approach that covers the entire image. For description step, we use SIFT descriptor due to the discrimination power it showed in the literature (Mikolajczyk & Schmid 2005). We applied PCA to de-correlate and reduce the SIFT descriptors to 64-D vectors as the correlation has large impact on the performance of the K-Means clustering (Coates & Ng 2012). Reducing the SIFT descriptors to 64-D vectors is the common practice applied in the literature (Rothacker 2011) (Rothacker et al. 2012). For codebook generation, we applied the K-Means clustering algorithm on a set consisting of one million descriptors selected randomly from the training samples. Moreover, Gaussian Mixture Models GMMs were estimated for the selected descriptors. As the codebook size is a crucial parameter for the BoF framework (Coates, Lee, et al. 2011), we gradually increased the codebook size as much as possible. We generated several codebooks of sizes in the range from 128 to 2048. For encoding, hard assignment and soft assignment based on the GMMs were implemented. The study of Coates and Ng (Coates & Ng 2012) showed that the K-Means clustering would achieve good performance with strong encoding schemes. Accordingly, in this section we applied soft assignment based on the GMM as this encoding scheme was applied in the literature of handwriting recognition (Rothacker 2011) (Rothacker et al. 2012). The image final feature vector was obtained by the average pooling (Section 2.5.6). The theoretical and experimental analysis of the pooling operations conducted by Boureau et al. (Boureau, Ponce, et al. 2010) (Boureau, Bach, et al. 2010) showed that the average pooling would achieved superior performance than max pooling for images with homogenous background

clutter. Since the images of handwritten text have clear background, our evaluation was restricted to the average pooling.

The system is implemented in MATLAB R2012b and run on a server with two Intel Xeon X5690 processors of 3.47 GHz and 88 GB RAM running Windows 7 Professional N. We used the *CornerDetector System object* of the MATLAB Computer Vision System Toolbox for Harris implementation and VLFeat library (Vedaldi & Fulkerson 2010) for the other algorithms.

Table 3.1: Configuration of the BoF framework baseline

Stage	Specification
Detector	Harris: region size of 24 pixels Harris-Laplace: the region scale specifies the region size. Dense Sampling: 4 grids of sizes 16, 24, 32 and 40 with 8-pixels stride
Descriptor	SIFT SIFT vectors are reduced to 64-D by applying PCA
Codebook Learning	K-Means applied on 1 million random descriptors GMM estimated over the codebook
Codebook Size	128, 256, 512, 1024 and 2048
Encoding	Hard Assignment Soft Assignment based on GMM

3.1.2. Experimental Results

To evaluate the framework baseline we conducted extensive experimentation on two public datasets viz., the non-touching Arabic Indian digits and non-touching Arabic subwords datasets of the CENPARMI database. The database was collected for Arabic checks processing at the Center for Pattern Recognition and Machine Intelligence, Concordia University, Canada in collaboration with Al-Rajhi Bank, Saudi Arabia (Al-Ohali et al. 2004). We start by describing the two datasets and the recognition systems, then we present and discuss the experimental results.

3.1.2.1. The Non-Touching Arabic Indian Digits Dataset

The non-touching Arabic/Indian digit dataset contains 10425 image samples of isolated Arabic/Indian digits (0-9) extracted from the courtesy amount field of the checks. The samples are partitioned into training set (7390 images) and test set (3035 images), where the samples of each digit are grouped into separate class in the training and test sets. Figure 3.1 shows the statistics of the digit classes with a sample image example from each digit class.

3.1.2.2. The Non-Touching Arabic Subwords Dataset

The non-touching Arabic subwords dataset contains 27985 image samples of Arabic subwords used in legal amounts of Arabic checks. The dataset has 87 classes corresponding to the Arabic subwords encountered in the legal amount filed of the collected checks. As the samples are extracted from real checks, the distribution of the samples of the classes are unbalanced, with some classes having a single sample in the training set while the test set is empty.

Two sets of experiments are conducted on the non-touching subwords dataset. The first is conducted on the most frequent 10 classes (Figure 3.2) in order to compare our results with published work (Pastor 2014), while in the second set we consider all the classes that contain at least one sample in the training set and one sample in the test set. There are 69 such classes in the datasets. Figure 3.3 shows useful statistics for these classes.

Class	Training	Testing	Total	Sample
0	3793	1574	5367	
1	782	304	1086	١
2	545	225	770	٢
3	362	144	506	٣
4	307	133	440	٤
5	649	263	912	٥
6	279	111	390	٦
7	233	109	342	٧
8	246	98	344	٨
9	194	74	268	٩
Total	**7390**	**3035**	**10425**	

Figure 3.1: Statistics of the non-touching digits dataset

No	Training	Testing	Total	Sample
1	2061	859	2920	
2	1896	781	2677	
3	1726	724	2450	
4	1301	529	1830	
5	1175	490	1665	ل
6	1159	521	1680	ا
7	1077	442	1519	
8	1005	410	1415	
9	961	396	1357	ن
10	745	304	1049	
Total	**13106**	**5456**	**18562**	

Figure 3.2: Statistics of the most frequent 10 classes of the non-touching subwords dataset

No	Code	Train	Test	Total	Sample
1	1-00	1993	829	2822	
2	1-05	4	2	6	
3	1-06	10	4	14	
4	1-07	24	8	32	
5	1-08	2061	859	2920	
6	1-09	567	244	811	
7	1-10	1175	490	1665	
8	1-11	961	396	1357	
9	1-12	242	103	345	
10	1-14	1896	781	2677	
11	1-16	37	16	53	
12	2-00	821	330	1151	
13	2-02	6	3	9	
14	2-03	187	82	269	
15	2-05	92	36	128	
16	2-06	62	31	93	
17	2-07	2	1	3	
18	2-08	81	30	111	
19	2-09	1726	724	2450	
20	2-10	745	304	1049	
21	2-11	216	83	299	
22	2-12	69	25	94	
23	2-13	1301	529	1830	
24	2-14	12	4	16	
25	3-00	51	20	71	
26	3-02	2	1	3	
27	3-04	161	61	222	
28	3-06	80	37	117	
29	3-07	4	1	5	
30	3-08	362	147	509	
31	3-09	319	121	440	
32	3-10	55	22	77	
33	3-11	11	5	16	
34	3-13	4	2	6	
35	3-14	11	4	15	

No	Code	Train	Test	Total	Sample
36	3-15	1	1	2	
37	3-17	114	47	161	
38	3-19	62	27	89	
39	3-20	23	11	34	
40	3-21	473	198	671	
41	3-22	1077	442	1519	
42	3-23	1005	410	1415	
43	3-24	130	49	179	
44	3-25	12	5	17	
45	3-27	2	1	3	
46	3-29	10	4	14	
47	3-31	71	28	99	
48	3-33	93	39	132	
49	3-35	1	1	2	
50	4-00	21	10	31	
51	4-01	83	35	118	
52	4-02	6	2	8	
53	4-04	64	26	90	
54	4-06	40	18	58	
55	4-09	13	5	18	
56	4-11	277	116	393	
57	4-13	158	55	213	
58	4-15	83	38	121	
59	4-17	69	33	102	
60	4-18	96	36	132	
61	4-19	4	1	5	
62	4-20	61	28	89	
63	4-21	33	15	48	
64	4-27	4	2	6	
65	5-02	87	29	116	
66	5-04	183	80	263	
67	5-05	10	4	14	
68	5-06	78	40	118	
69	5-10	3	1	4	

Figure 3.3: Statistics of the non-touching subwords dataset

3.1.2.3. The Handwriting Recognition System

To assess the quality of the learned features, the framework baseline is integrated with a holistic handwriting recognition system. Below, we present the implementation details of the system.

In the preprocessing step, image samples are normalized to 64 pixels in height while maintaining the aspect ratio. Height normalization is a common preprocessing step for reducing the side effect of the variety in text size. 64-pixel height normalization was used with the Non-Touching Digits Dataset in (Mahmoud 2009) and (Mahmoud & Al-Khatib 2010). For feature extraction, we applied the BoF framework as presented in Section 3.1.1. The classification step of our recognition system is implemented using Support Vector Machine (SVM) with linear kernel.

3.1.2.4. Non-Touching Digits Dataset Performance

The recognition accuracies obtained on the non-touching Arabic/Indian digits are shown in Figure 3.4. The results show that, on average, dense sampling outperforms the two interest point detectors. This result is consistent with comparative studies conducted for visual object recognition tasks that show that dense sampling produces better recognition rates than interest point detectors (Nowak et al. 2006). The best results of the three sampling strategies are shown in Table 3.2.

Comparing the hard and soft assignments, we found that soft assignments gave better accuracies in all cases, since the soft assignment reduces the side effect of quantization distortion associated with the hard assignment. The results show also the positive effect of increasing the codebook size up to 2048. Best recognition rate of 99.34% was achieved by

dense sampling with soft quantization and a codebook size of 2048. The hard quantization with the codebook of the same size achieved 99.11%, while codebook size of 1024 achieved 99.14% and 99.05% with the soft and hard assignments respectively. Similarly, codebook size of 512 achieved a 99.05% recognition rate with soft assignment. These results outperformed the state-of-the-art recognition rates published in the literature on the same dataset. Table 3.3 compares our results with the results in the literature. The table shows the statistical significance of the recognition rates at the 95% confidence level. The table shows that some of the results are not statistically significant, although they are higher than published work. This indicates the power of the framework in learning robust feature representation.

Figure 3.5 shows the confusion matrix of the tested samples for the configuration that gives the best accuracy (dense sampling with soft quantization and codebook size of 2048). We achieved a recognition rate of 100% in six classes (2, 3, 6, 7, 8 and 9), while most of the misclassified samples are in class 0 (10 samples) and class 1 (7 samples). Twenty total samples were misclassified. These samples are shown in Figure 3.6. It is clear that the misclassified samples of classes 0 and 1 are similar, making the differentiation between them hard even for a human. The large, wide hole in sample #10 makes digit 0 looks like 5. On the other hand, the thick border and small hole in sample #19 make digit 5 looks like 0. The soft mid-concavity of the misclassified sample of class 4 (sample #18) makes it similar to digit 2, especially when we compare it with some training samples of digit 2 that ended with sharp right tail as in (ﺼ) and (ﺼ). Our approach successfully recognized several challenging samples in this class, like (ﺼ), (ﺼ), (ﺼ) and (ﺼ) that were misclassified in other works (Mahmoud & Al-Khatib 2010) (Awaida & Mahmoud 2014).

Our approach also tolerates the cut found in the middle of some samples like the ones in (١), (٨) and (٦). Digit 3 is usually written with three upper strokes (٣) and sometimes with two (٢). While previous approaches (Mahmoud & Al-Khatib 2010) (Awaida & Mahmoud 2014) have difficulties to recognize them, our approach achieved 100% accuracy on this class.

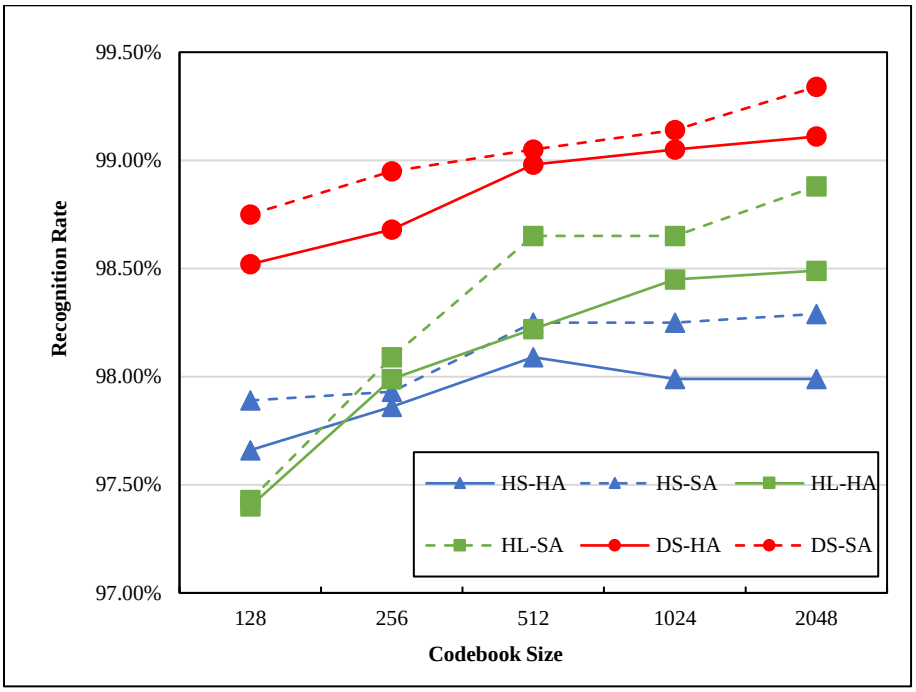


Figure 3.4: Recognition rates for digit dataset with different local feature detection and quantization approaches

HS = Harris, HL = Harris-Laplace, DS = Dense Sampling
HA = Hard Assignment, SA = Soft Assignment

Table 3.2: Best recognition accuracy achieved by the three detection methods

Codebook size = 2048, quantization = soft assignment

Class	Harris	Harris-Laplace	Dense Sampling
0	99.56%	99.68%	99.36%
1	96.71%	98.03%	97.70%
2	98.22%	97.78%	100.00%
3	96.53%	98.61%	100.00%
4	98.50%	97.74%	99.25%
5	96.21%	98.48%	99.24%
6	98.20%	100.00%	100.00%
7	93.58%	97.25%	100.00%
8	98.99%	97.98%	100.00%
9	94.59%	94.59%	100.00%
Average	98.29%	98.88%	99.34%

Table 3.3: Recognition accuracies reported in the literature for CIMPARMI non-touching digits dataset vs. the best accuracies achieved in this work

Features and Classifier	Accuracy	Statistical significance
Dense Sampling, 2048 codebook, Soft Assignment with SVM (This work)	99.34%	±0.29
Dense Sampling, 1024 codebook, Soft Assignment with SVM (This work)	99.14%	±0.32
Dense Sampling, 2048 codebook, Hard Assignment with SVM (This work)	99.11%	±0.33
Dense Sampling, 1024 codebook, Hard Assignment with SVM (This work)	99.05%	±0.34
Dense Sampling, 512 codebook, Soft Assignment with SVM (This work)	99.05%	±0.34
GSC Features with SVM (Awaida & Mahmoud 2014)	99.04%	±0.34
Log-Gabor Filter Response Features with SVM (Mahmoud & Al-Khatib 2010)	98.95%	±0.35
Pixel Intensity Values with Bernoulli Mixtures (Romero et al. 2007)	98.10%	±0.45
Pixel Intensity Values with Bernoulli HMM (Giménez et al. 2011)	98.00%	±0.46
Spatial Gabor Filter Response Features with 1-NN (Mahmoud 2009)	97.99%	±0.46

Actual Class	Predicted Class										Recognition Rate
	0	1	2	3	4	5	6	7	8	9	
0	1564	8	0	0	1	1	0	0	0	0	99.36%
1	7	297	0	0	0	0	0	0	0	0	97.70%
2	0	0	225	0	0	0	0	0	0	0	100.00%
3	0	0	0	144	0	0	0	0	0	0	100.00%
4	0	0	1	0	132	0	0	0	0	0	99.25%
5	1	0	0	0	0	262	0	1	0	0	99.24%
6	0	0	0	0	0	0	111	0	0	0	100.00%
7	0	0	0	0	0	0	0	109	0	0	100.00%
8	0	0	0	0	0	0	0	0	99	0	100.00%
9	0	0	0	0	0	0	0	0	0	74	100.00%
										Average	**99.34%**

Figure 3.5: Confusion matrix of the tested digit samples for dense sampling with codebook of size 2048 and soft quantization

No.	Image	Actual Class	Predicted Class	No.	Image	Actual Class	Predicted Class
1		0	1	11		1	0
2		0	1	12		1	0
3		0	1	13		1	0
4		0	1	14		1	0
5		0	1	15		1	0
6		0	1	16		1	0
7		0	1	17		1	0
8		0	1	18		4	2
9		0	4	19		5	0
10		0	5	20		5	7

Figure 3.6: Images of the misclassified digit samples

3.1.2.5. Non-touching Subwords Dataset Performance

The recognition accuracies obtained on the most frequent 10 classes and all classes that contain at least one sample in the training set and one sample in the testing set are shown in Figure 3.7. In all experiments, SIFT descriptors are extracted densely using four scales viz 4, 6, 8 and 10 pixels. Five codebooks of sizes 128, 256, 512, 1024 and 2048 are used. Both the Hard and Soft Assignments are evaluated.

Similar to the digit dataset, the soft assignment outperformed the hard assignment in all codebook sizes, and the best accuracies are achieved with a codebook of size 2048. In the most frequent 10 classes, best recognition rate of 95.53% (±0.48) was achieved by the soft assignment with the 2048 codebook. This result is statistically significant and about 6.4% better than the result reported in (Pastor 2014), where a recognition rate of 89.10% (±0.71) was reported on the same classes using pixel intensity values and Bernoulli HMM. Figure 3.8 shows the confusion matrix of the tested samples for the configuration that gives the best result (soft quantization and codebook of size 2048). The lowest accuracy was for the NOON (ن). This is attributed to the large variety of the writing style of this letter (Figure 3.9). The two letters RAA (ر) and WAW (و) have similar writing styles. We noticed that the confusion between these two classes constitutes about 36.5% of the total misclassification errors. Other misclassified samples have challenging writing styles that made them similar to other classes. In Figure 3.9, we show samples of such challenging styles.

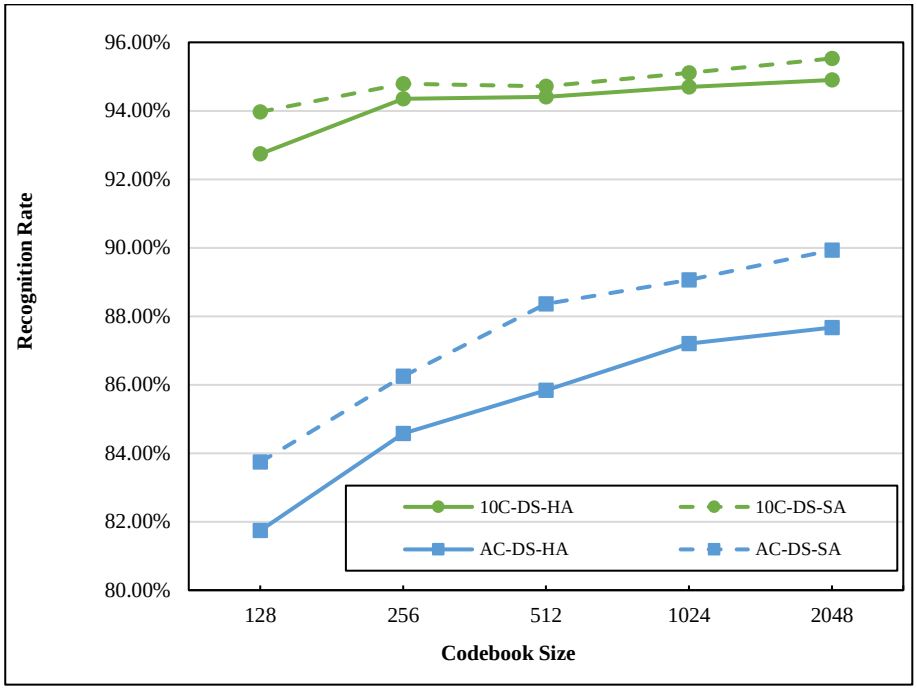


Figure 3.7: Recognition rates for the non-touching subwords dataset

10C = The Most frequent 10 classes, AC = All Classes

HA = Hard Assignment, SA = Soft Assignment

	Predicted Class										Recognition Rate
	ا	ر	ل	ن	و	لا	ف	ي	نقطة	مقطع	
ا	499	15	2	0	2	2	0	1	0	0	95.78%
ر	4	804	2	8	39	2	0	0	0	0	93.60%
ل	5	4	466	6	0	4	1	2	2	0	95.10%
ن	0	7	7	366	2	5	4	3	2	0	92.42%
و	0	50	2	0	722	5	0	0	2	0	92.45%
لا	0	2	5	1	2	710	2	1	0	1	98.07%
ف	0	0	2	2	0	3	297	0	0	0	97.70%
ي	1	0	1	2	0	5	0	518	2	0	97.92%
نقطة	0	0	0	12	1	4	0	3	421	1	95.25%
مقطع	0	0	0	0	0	0	0	0	1	409	99.76%
										Average	**95.53%**

Figure 3.8: Confusion matrix of the tested samples of the most frequent 10 classes of the non-touching subwords dataset

No.	Image	Actual Class	Predicted Class	No.	Image	Actual Class	Predicted Class
1		ا	ر	11		و	ل
2		ا	و	12		لا	ل
3		ر	و	13		لا	ف
4		ر	ا	14		لا	يا
5		ل	ا	15		ف	مقط
6		ن	نـ	16		ف	ن
7		ن	ر	17		يا	لا
8		ن	لا	18		نـ	ن
9		ن	و	19		نـ	و
10		و	ر	20		مقط	نـ

Figure 3.9: Examples of misclassified subwords samples in the most frequent 10 classes of the non-touching subwords dataset

For the complete subwords dataset, the best accuracy we achieved was 89.93% (±0.56) with Soft Assignment and 2048 codewords. These results are encouraging on such a challenging dataset containing large number of classes, with many classes of few samples. This is still better and statistically significant than the recognition accuracy reported in (Pastor 2014) for the most frequent 10 classes. Figure 3.10 shows the per-class accuracy using Soft Assignment and 2048 codewords. The errors are attributed to the lack of training samples, as the classes with less than five training samples (e.g., classes #2, 17, 29, 36, and 61 ((ـــــے), (ـــــــہ), (ـــــح), (ـــــــ), and (ـــــــہ)) have poor performance. The classes having similar writing style, e.g., classes #4 and 5 ((د) and (ر)), classes #6, 8 and 10 ((ـــــــق), (ـــــو) and (ن)) and classes #40 and 41 ((ـــــــــــ) and (ـــــــــــ)) have large confusion. Class# 8 (ن) has several writing styles, so samples from other classes with unusual writing styles are sometimes assigned to this class. Figure 3.11 shows examples of challenging samples. Despite that, we have achieved high accuracy (above 90%) in many classes containing several hundred test samples, like Classes #1, 19, 20, 23 and 42. Some samples in these classes have complicated writing styles which are hard for humans to recognize without context (Figure 3.12).

No	#Samples	Correctly Classified	Mis-classified	Accuracy	No	#Samples	Correctly Classified	Mis-classified	Accuracy
1	829	807	22	97.35%	36	1	0	1	0.00%
2	2	0	2	0.00%	37	47	44	3	93.62%
3	4	2	2	50.00%	38	27	21	6	77.78%
4	8	3	5	37.50%	39	11	8	3	72.73%
5	859	797	62	92.78%	40	198	189	9	95.45%
6	244	217	27	88.93%	41	442	397	45	89.82%
7	490	459	31	93.67%	42	410	406	4	99.02%
8	396	341	55	86.11%	43	49	40	9	81.63%
9	103	83	20	80.58%	44	5	1	4	20.00%
10	792	721	71	91.04%	45	1	0	1	0.00%
11	5	1	4	20.00%	46	4	0	4	0.00%
12	330	278	52	84.24%	47	28	15	13	53.57%
13	3	1	2	33.33%	48	39	21	18	53.85%
14	82	31	51	37.80%	49	1	0	1	0.00%
15	36	24	12	66.67%	50	10	3	7	30.00%
16	31	24	7	77.42%	51	35	28	7	80.00%
17	1	0	1	0.00%	52	2	0	2	0.00%
18	30	25	5	83.33%	53	26	9	17	34.62%
19	724	706	18	97.51%	54	18	14	4	77.78%
20	304	291	13	95.72%	55	5	0	5	0.00%
21	83	76	7	91.57%	56	116	106	10	91.38%
22	25	13	12	52.00%	57	55	48	7	87.27%
23	529	508	21	96.03%	58	38	24	14	63.16%
24	4	0	4	0.00%	59	33	21	12	63.64%
25	20	13	7	65.00%	60	36	31	5	86.11%
26	1	1	0	100.00%	61	1	0	1	0.00%
27	61	52	9	85.25%	62	28	20	8	71.43%
28	37	29	8	78.38%	63	15	8	7	53.33%
29	1	0	1	0.00%	64	2	0	2	0.00%
30	147	138	9	93.88%	65	29	24	5	82.76%
31	121	100	21	82.64%	66	80	77	3	96.25%
32	22	19	3	86.36%	67	4	0	4	0.00%
33	5	2	3	40.00%	68	40	32	8	80.00%
34	2	0	2	0.00%	69	1	0	1	0.00%
35	4	0	4	0.00%					
Total Samples: 8172; Correctly Classified: 7349; Misclassified: 823; Average Accuracy: 89.93%									

Figure 3.10: Per-class accuracy achieved by soft assignment and 2048 codewords on the complete subwords dataset

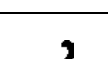
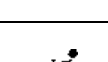
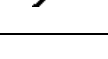
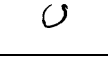
No.	Image	Actual Class	Predicted Class	No.	Image	Actual Class	Predicted Class
1		ت	ن	11		تنه	نيف
2		د	ر	12		ثنا	لفا
3		د	و	13		لفاً	لف
4		ر	و	14		تنى	ئة
5		و	ر	15		ثما	ن
6		ن	ف	16		ثنين	نيف
7		ف	ن	17		نيف	عشر
8		ي	نيف	18		فقط	لا
9		ين	ن	19		مئة	ن
10		ين	ن	20		تسعة	سبعة

Figure 3.11: Examples of misclassified subwords samples of the complete subwords dataset

No.	Image	Class	No.	Image	Class
1		ا	6		مقطع
2		لا	7		مقطع
3		لا	8		مقطع
4		يا	9		لف
5		يا	10		لف

Figure 3.12: Examples of challenging samples that were correctly recognized

3.2. Utilizing the Characteristics of Arabic Handwritten Text in Feature Learning Framework

We believe that utilizing the characteristics of the handwritten text could enhance the quality of the learned representation and reduce the overheads involved in several stages of the framework. In this section we show how the characteristics of the handwritten text could be employed in improving SIFT descriptors.

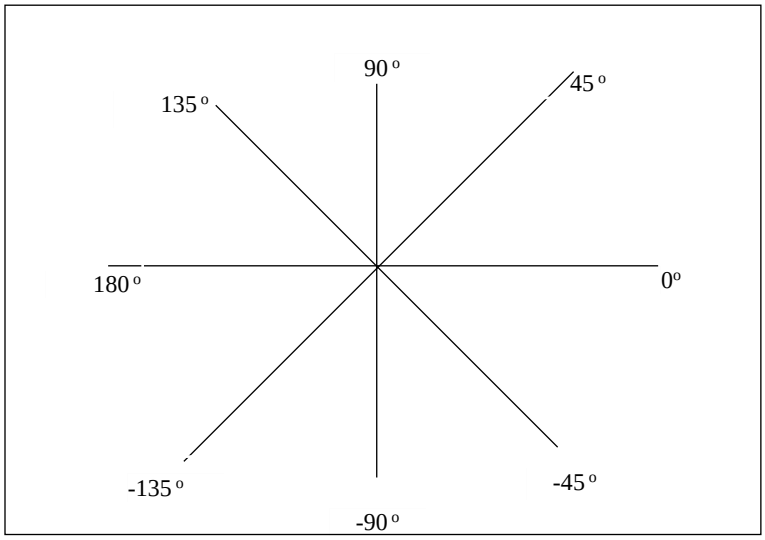
SIFT is the de-facto standard for feature learning frameworks in computer vision. Despite its discriminative power, SIFT has two main drawbacks. The first is the computational overhead due to the extensive computations involved in evaluating the gradient magnitude and orientation and the associated pre-smoothing step necessary for improving the gradient quality. The second is the high dimensionality of the resulting feature vector. This motivated researchers looking for other approaches that could achieve the performance of the gradient magnitude and orientation at low cost as well as to reduce the size of the descriptor vector. The SURF descriptor (Bay et al. 2008) used Haar wavelet response that are computed efficiently using integral images as an approximation to the gradient magnitude and orientation. The CS-LBP descriptor (Heikkilä et al. 2009) replaced the gradient information by the response of the Local Binary Pattern (LBP) that is computationally efficient. In the same manner the CS-LTP (Gupta et al. 2010) and WOS-LTP (Huang et al. 2015) descriptors both used the response of the extended LBP operator named Local Ternary Pattern (LTP). It is worthily to note that the LBP and its extension LTP are closely related to the gradient as these operators essentially evaluate the pixel intensity differences. Instead of continuing in evaluating the gradient magnitude and orientation values, they used only the sign of the differences. To cope with the large

dimensionality of the descriptor vector, several approaches were proposed. One of the earliest approaches is the PCA-SIFT (Ke & Sukthankar 2004) that achieved the discrimination power of SIFT with descriptors of 20 to 36 elements by applying PCA on the gradient magnitudes. The SURF algorithm (Bay et al. 2008) produces a descriptor of 64 elements by computing 4 bins in each of the 16 regions, instead of the 8 bins used in SIFT. In the following two subsections we present our proposed approaches for reducing the dimensionality of SIFT descriptors and for speeding up the computation of the gradient magnitude and orientation by utilizing the characteristics of the handwritten text.

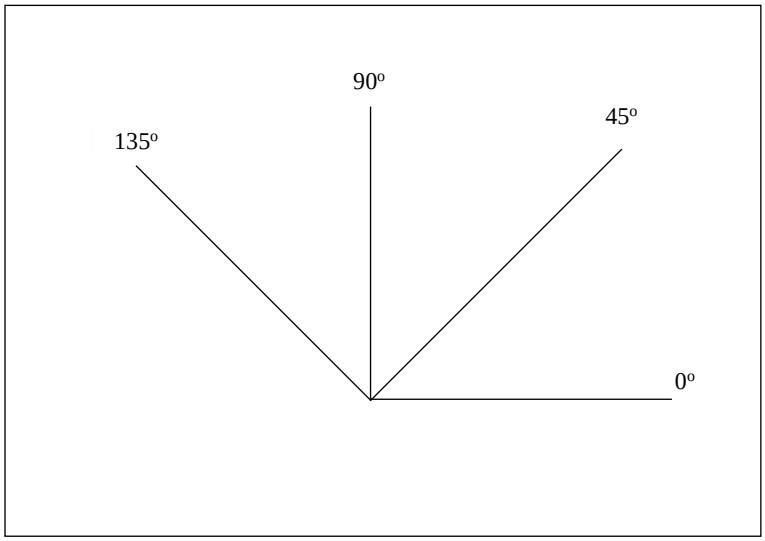
3.2.1. Reducing Descriptor Dimensionality

SIFT descriptor relies on the distribution of the gradient magnitude to describe the regions of interest. Once the pixels' gradient magnitude and orientation are computed, the patch spatial area is divided into 4×4 regions and the 360-degree gradient orientation range is quantized into 8 orientation bins (Figure 3.13 (a)). The histogram of the gradient magnitudes at each region are computed, giving up a descriptor vector of 4×4×8 = 128 elements for the patch. In text recognition, however, our concern is the text orientation regardless of the pixels orientation. For instance, when a pixel shows -90^{o} orientation, this indicates that the pixel lies on the lower edge of a horizontal text whereas $+90^{o}$ orientation indicates that it lies on the upper edge of a horizontal text. In both cases the text is horizontal (see Figure 3.14). Therefore, the two symmetric orientation bins can be combined into a single bin, giving 4 orientation bins instead of 8 (Figure 3.13 (b)) and the descriptor dimensionality is reduced from 128 to 64. Note that the distance between the two adjacent bins is still 45^{o} similar to the original SIFT algorithm. The only difference is that the value of the negative orientation bins are accumulated to the corresponding

symmetric positive orientations as the orientation sign is insignificant in describing the direction of the text lines. This modification produces shorter vectors of the same discriminative power of the original SIFT.

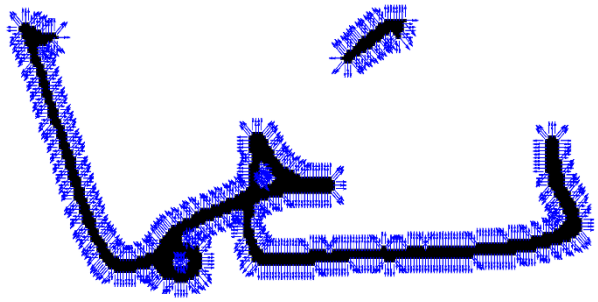


(a) The range [0^O , 360^O] is quantized into 8 bins

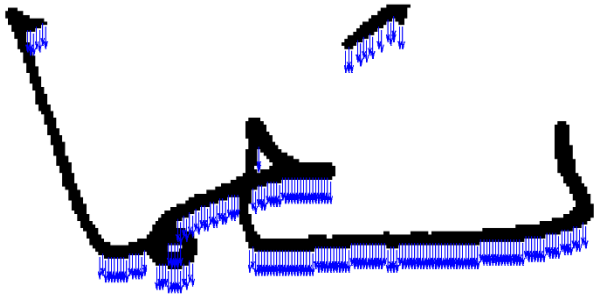


(b) Combining the symmetric orientation bins gives 4 bins

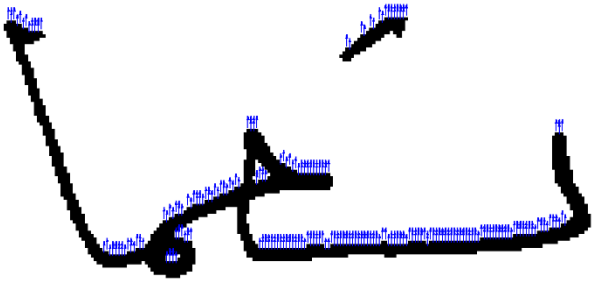
Figure 3.13: Orientation Quantization



(a) Gradient orientation computed for each pixel



(b) -90º orientation indicates that the pixel lies on the lower edge of a horizontal text



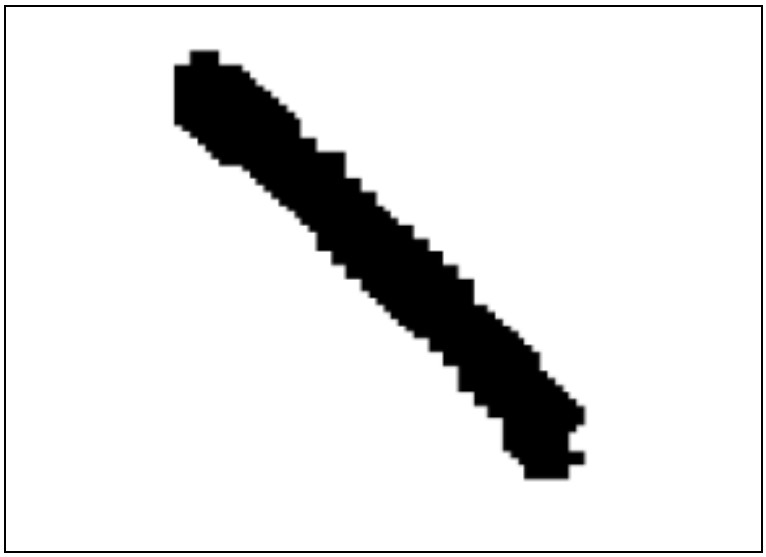
(c) +90º orientation indicates that the pixel lies on the upper edge of a horizontal text

Figure 3.14: Pixel orientation vs. text orientation

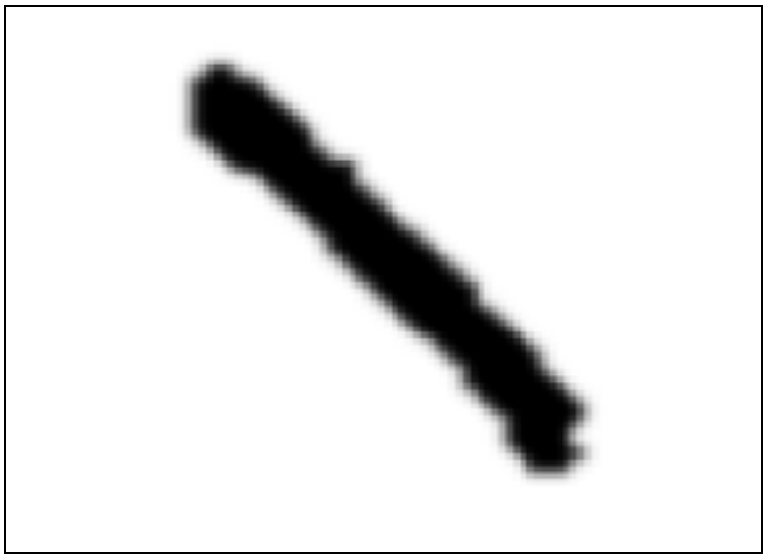
3.2.2. Fast Computation of Gradient Magnitude and Orientation

In order to compute the pixels' gradient magnitude and orientation, the original SIFT algorithm (Lowe 2004) applies the basic derivative filters (h_x = [-1 0 1] , h_y = [-1 0 1]^T) for evaluating the horizontal and vertical derivatives (d_x , d_y) at each pixel. The justification of using the basic derivative filters rather than large filters like Prewitt or Sobel filters is that the image has to be pre-smoothed by Gaussian kernels in the previous stages of interest regions detecting. The enhanced filter like Prewitt or Sobel take derivative in one direction and smooth in the orthogonal direction (Young et al. 1998). Preceding the basic derivative filters by Gaussian smoothing makes them achieving the performance of other enhanced filters, e.g., Prewitt and Sobel. Implementations that applied SIFT on dense sampling like VLFeat dsift have to pre-smooth the image by Gaussian kernel proportional to the descriptor spatial area in order to make the descriptor less sensitive to noise.

Gaussian smoothing is computationally expensive. Several descriptors like the SURF (Bay et al. 2008), BRIEF (Calonder et al. 2012) and LDP (Yang & Cheng 2014) approximate the Gaussian smoothing by box filters that could be efficiently implemented by integral images. In the case of handwritten text where the images are binary, the noise associated with this type of images is due to the writing style and the binarization algorithm. The noise appears as sharp edges due to the transition from 0 to 1 or from 1 to 0. Applying Gaussian smoothing would attenuate these edges rather than permanently eliminate the notches on the text borders (Figure 3.15). Moreover, smoothing affects the (binary) values of almost all pixels even those pixels that are far from the edges of the text line. The changes in the values of the pixels in the middle of the text line generate noisy gradients.



(a) Original image



(b) Smoothed image

Figure 3.15: A Digit Sample smoothed by a Gaussian kernel

The original digit has 821 pixels of value 0. After smoothing, only 8 pixels remained 0

To avoid these effects, we eliminated the smoothing step and applied the basic derivative filters directly on the text images. Since the images are binary, the filter response at any image pixel would take one out of three values {-1 , 0 , 1} based on the intensity value (0/1) of the left/right and top/bottom neighbor pixels. Consequently, the computation of gradient magnitude and orientation can be computed by building lookup tables instead of applying computationally expensive procedures for computing arctan and square-root functions. The proposed lookup tables are shown in Figure 3.16 (a and b).

The possible values of the gradient orientation in Figure 3.16 (b) are exactly the 8 orientation bins shown in Figure 3.13 that implies the gradient magnitude would be accumulated to exactly one bin according to its orientation. Utilizing these lookup tables would enable us to construct faster SIFT descriptor comparing to the original algorithm. Using the reduction approach explained in the previous subsection, the descriptor dimensionality could be reduced to half by combining the corresponding symmetric orientation bins. The proposed lookup table for computing pixel orientation after combining the corresponding symmetric orientations is shown in Figure 3.16 (c).

d_y	d_x		
	-1	**0**	**1**
-1	$\sqrt{2}$	1	$\sqrt{2}$
0	1	0	1
1	$\sqrt{2}$	1	$\sqrt{2}$

(a) gradient magnitude

d_y	d_x		
	-1	**0**	**1**
-1	-135º	-90 º	-45 º
0	+180 º	+90 º	0 º
1	+135 º	+90 º	+45 º

(b) gradient orientation before combining the symmetric orientation bins

d_y	d_x		
	-1	**0**	**1**
-1	+45º	+90 º	+135º
0	0 º	+90 º	0 º
1	+135 º	+90 º	+45 º

(c) gradient orientation after combining the symmetric orientation bins

Figure 3.16: The proposed lookup tables for computing gradient magnitude and orientation

3.2.3. Experimental Results

Based on the modifications presented above, we have implemented two novel versions of SIFT, named *unsigned-SIFT* and *binary-SIFT*. *Unsigned-SIFT* smooths the input image by a Gaussian kernel proportional to the descriptor spatial area similar to the one used by the VLFeat *phow*. Gradient magnitudes and orientations are computed using the same procedures as VLFeat *dsift*. However, instead of generating 8 bins in each of the 4×4 spatial regions, the contributions of the corresponding symmetric orientation bins are accumulated, giving 4 bins per region. Accordingly, *unsigned-SIFT* produces a 64-D descriptor for the input patch. In *binary-SIFT*, the Gaussian smoothing step is eliminated and the gradient magnitude and orientation are computed for the binary image using the lookup tables shown in Figure 3.16. Similar to *Unsigned-SIFT*, the contributions of the corresponding symmetric orientation bins within each spatial region are accumulated, giving a 64-D descriptor for the patch. The two versions are implemented based on the open-source VLFeat library by modifying *dsift* command, which is an efficient implementation for extracting dense SIFT descriptors for gray-scale images.

To evaluate the performance of the novel versions, we integrated them with our feature learning framework and conducted several experiments on the non-touching Arabic Indian digit and non-touching Arabic subwords datasets. In these experiments, we used dense sampling with four patch sizes (16, 24, 32 and 40 pixels) and a stride of 2 pixels in the four grids. The original SIFT as well as the two novel versions were applied in the description step. The obtained descriptors are de-correlated by applying PCA. Five codebooks of sizes 128, 256, 512, 1024 and 2048 are generated by k-means clustering, and the hard assignment

is utilized in the quantization step. Figure 3.17 compares the recognition accuracies of the three SIFT versions.

The results show that *unsigned-SIFT* and *binary-SIFT* achieve comparable performance to the original SIFT, although their descriptors have only 64 elements. The two versions achieved better performance with larger codebooks in the digits and the most frequent 10 sub-word classes. In the complete sub-words dataset, however, the *binary-SIFT* was the worse. This might be attributed to the lack of enough training samples. In addition to the promising recognition accuracies, the two versions take less time to compute. Due to their lower dimensionality, the clustering and quantization became faster. Table 3.4 shows the CPU times of computing the descriptors of the non-touching Arabic Indian digit dataset using the original SIFT and the two novel versions. We show also the CPU times of clustering three sets, each with one million random descriptors generated by one of the three versions into 1024 clusters. The two versions achieved up to 2.16× speedup in description generation and clustering steps, which indicates that utilizing the characteristics of the handwritten text has reduced the computational overhead.

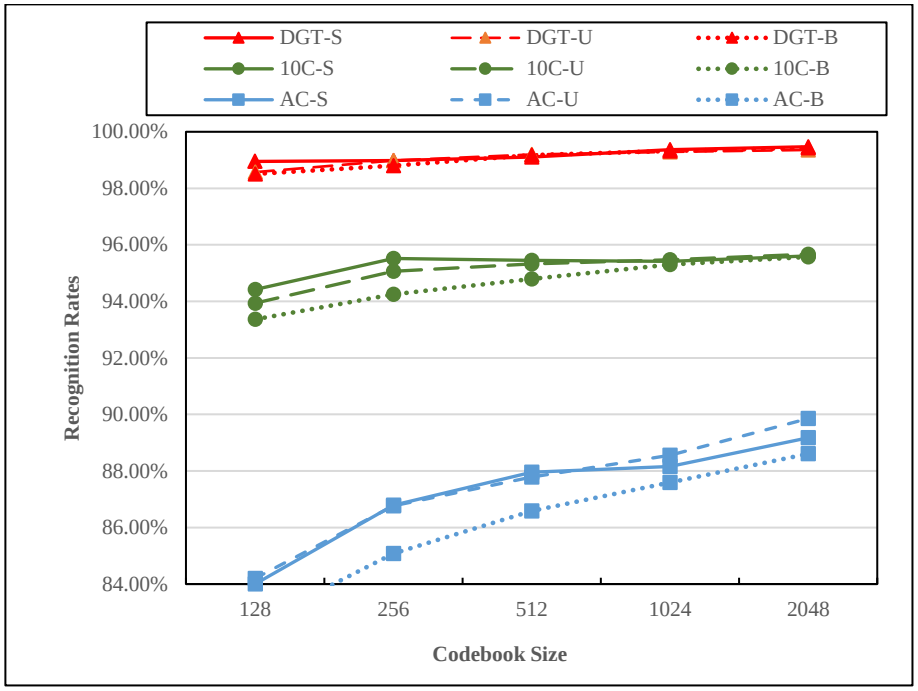


Figure 3.17: Comparing the Recognition accuracies of the three versions of SIFT

DGT = Digits dataset, 10C = Most frequent 10 classes, AC = All Classes
S = Original SIFT, U = Unsigned-SIFT, B = Binary-SIFT

Table 3.4: CPU time (in milliseconds) elapsed in computing the descriptors of the non-touching Arabic Indian digit dataset and in clustering 1 million descriptors into 1024 clusters

	SIFT	**Unsigned-SIFT**	**Binary-SIFT**
Descriptors Computation	211.20	121.43	97.54
Clustering	507.13×10^3	341.07×10^3	356.37×10^3

3.3. Bag-of-Features Framework with Gabor Filters Features

The two-stage BoF framework utilizes the discriminative low-level features in producing robust image representation. Though the first stage of the BoF framework could be implemented by any type of low-level features, most of the works that applied BoF relied on gradient features in the SIFT format. The main reason for this is that several comparative studies showed that SIFT outperforms existing local descriptors (Mikolajczyk & Schmid 2005) (Hu et al. 2015). In this section we investigate the performance of other types of appearance-based low-level features that have shown superior performance in handwriting recognition.

The Gabor filters features are statistical measurements extracted from the Gabor Filters Response (See Section 2.6). We arrange the Gabor filter features into two different layouts, coined the *Statistical Gabor Features* (SGF) and *Gabor Descriptors* (GD).

3.3.1. Statistical Gabor Features (SGF)

The statistical Gabor Features (SGF) is the format that was applied in the previous works (Mahmoud 2008) (Mahmoud 2009) (Mahmoud & Al-Khatib 2010). A text image is sampled horizontally into 8-pixels-width samples and each sample is divided vertically into 8 regions. The mean and variance of the amplitude responses within each region are taken as the features of the region. Therefore, each image sample gives a feature vector of 8×2=16 elements for each scale and orientation in the Gabor filter bank. For 3 scales and 6 orientations per scale, we get 3×6=18 vectors -each of 16 elements- for each image sample. The Statistical Gabor Features for an image sample is a 288-d (16×18) vector obtained by concatenating the 18 feature vectors.

3.3.2. Gabor Descriptors (GD)

Gabor Descriptors produces features similar to the features produced by common visual descriptors like SIFT (Lowe 2004) and SURF (Bay et al. 2008) as the characteristics of this layout enables them to simulate the behavior of the human visual system (Lowe 2004). Gabor descriptor is constructed by sampling the image into square patches of P×P pixels. Each patch is divided into 2×2 regions. The amplitude responses of the different Gabor filters within each region are pooled using a statistical aggregation function. Four such functions are evaluated, viz., the max, sum, mean and variance. For 3 scales and 6 orientations per scale, the Gabor descriptor for a patch is a 72-d (2×2×3×6) vector. For a text image sample, the Gabor descriptors are extracted in two configurations: single-scale and multi-scale. In the single-scale configuration, the text image is sampled densely into 8×8 patches with a stride of 8 pixels. Each patch gives a 72-d Gabor descriptor. For multi-scale configuration, the text image is repeatedly sampled into patches of different scales. The 4-scale configuration we evaluated uses four patch sizes (viz. 8, 16, 24 and 32) with a stride of 8 pixels in the four scales.

3.3.3. Experimental Results

To evaluate the effectiveness of Gabor features, we apply them to the feature learning framework instead of 128-D SIFT descriptors. The obtained features are adapted to handwritten text recognition and extensive experimentations were conducted on the non-touching Arabic subwords dataset of CENPARMI database.

3.3.3.1. Evaluating Statistical Gabor Features

To evaluate the Statistical Gabor Features (SGF), we test two different Gabor filter banks. The first consists of one scale of wavelength of 3 and six orientations (0, 30, 60, 90, 120

and 150) while the second consists of three scales (wavelengths of 3, 6 and 12) and six orientations (0^o, 30^o, 60^o, 90^o, 120^o and 150^o) per scale. The 6 orientations were applied earlier for digit recognition (Mahmoud 2008) (Mahmoud & Al-Khatib 2010) as well as in texture image retrieval (Manjunath & Ma 1996). We applied several scales to provide scale-invariant representations.

Experiments with One Scale and Six Orientations

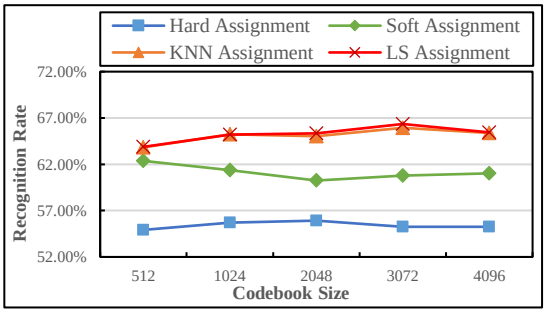
We conducted several experiments to evaluate the efficiency of the orientations we choose. In these experiments, each image in the training and test sets is filtered by a Gabor filter bank of one scale and six orientations. This gives six Statistical Gabor Feature vectors - each of size 16 elements- for each sample in the image. The 16–D feature vectors corresponding to each orientation are applied individually to our feature learning framework instead of the 128-D SIFT descriptors. Table 3.5 shows the configuration of the BoF framework applied in these experiments. We generated codebooks of large sizes (up to 4096 codewords) to provide more discrimination, since the non-touching Arabic subwords dataset has large number of classes. We also evaluated the K-Nearest-Neighbor Assignment (KNN Assignment) and Localized Soft-Assignment (LS Assignment) encodings (See Section 2.5.5), in addition to the Hard and Soft Assignments applied in Section 3.1.2. The two encoding schemes assign a local feature descriptor to a set of the closest codewords, in contrast to the GMM-based Soft Assignment that assigns to the whole codewords. These encoding schemes achieved good performance in visual object and scene images (Liu et al. 2011) (Avila et al. 2013). Our aim here is to evaluate them in handwriting recognition.

Table 3.5: Configuration of BoF framework applied in evaluating Statistical Gabor Features of individual orientation

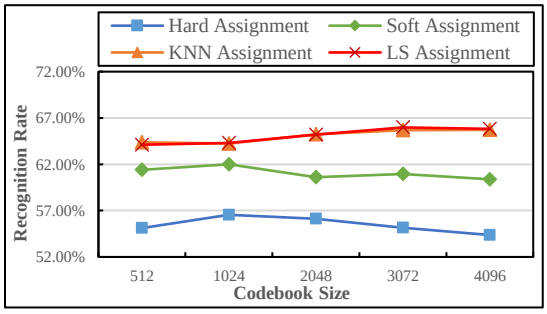
Parameter	Specification
Detector	Dense Sampling An image is horizontally sampled into 8-pixels-width samples
Descriptor	Statistical Gabor Features
Codebook Learning	K-Means applied on 1 million random descriptors GMM estimated over the codebook
Codebook Size	512 ,1024 ,2048 ,3072, 4096
Encoding	Hard Assignment Soft Assignment based on Gaussian Mixture Models (GMM) K-Nearest-Neighbor Assignment Localized Soft-Assignment

Figure 3.18 shows the recognition rates achieved by the Statistical Gabor Features of individual orientation. The results showed that for all the orientations, the KNN Assignment and LS Assignment encodings produce comparable results which outperform the two other encoding schemes (i.e., the Hard Assignments and Soft Assignments). The KNN Assignment and LS Assignment encoding schemes get the benefits of larger codebooks while the performance of the Hard Assignment and Soft Assignment dropped down when the codebook size went beyond 2048. This indicates that to get the beneficial of sophisticated soft assignments it is important to generate codebooks of large sizes.

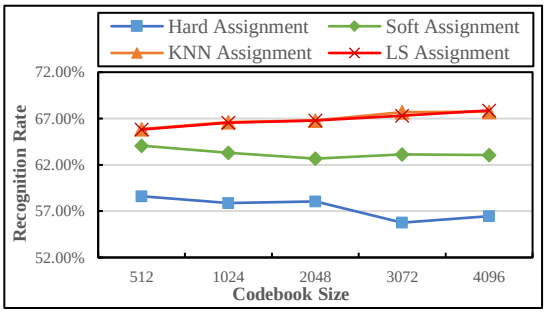
The best results achieved by each orientation are shown in Table 3.6. As shown in the table, the Statistical Gabor Features of the third orientation (60^o) achieved the best recognition rate (67.87%) followed by the sixth orientation (150^o) that achieved 67.50%. This is attributed to the property of the Gabor filter that the filter responds to the variation towards its orientation. For handwritten text, the variation in the horizontal and vertical directions is high due to the fact that most of the text components are horizontal or vertical. While the filters with horizontal and vertical orientations strongly respond to the variations in one orientation, the filters with diagonal orientations like 60^o and 150^o can observed discriminative variations in both directions. This explains the higher recognition accuracies achieved by the Gabor filters with the diagonal orientations 60^o and 150^o. Figure 3.19 visualizes the responses of the six filters on a sample sub-word image.



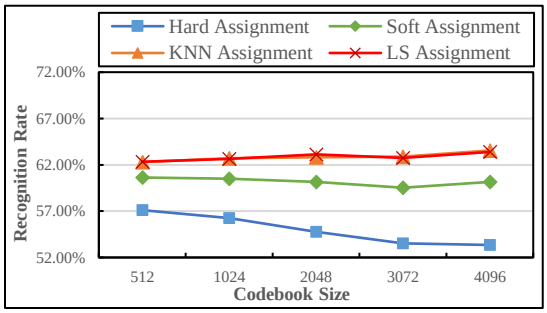
(a) Orientation 1 (0º)



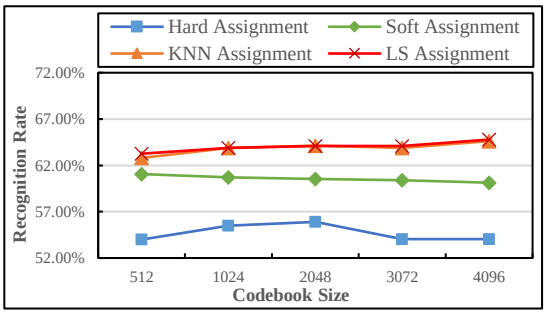
(b) Orientation 2 (30º)



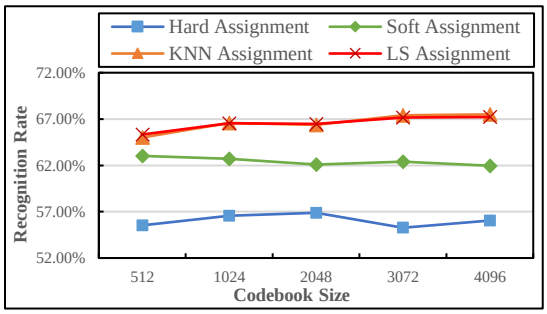
(c) Orientation 3 (60º)



(d) Orientation 4 (90º)



(e) Orientation 5 (120º)

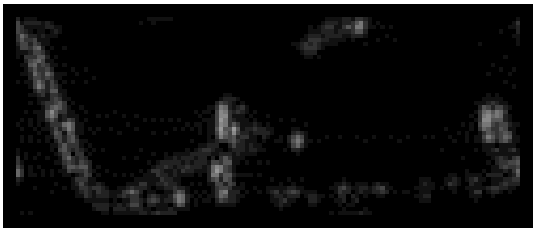


(f) Orientation 6 (150º)

Figure 3.18: Recognition rates for on non-touching Arabic sub-words with Statistical Gabor Features using individual orientations of single scale

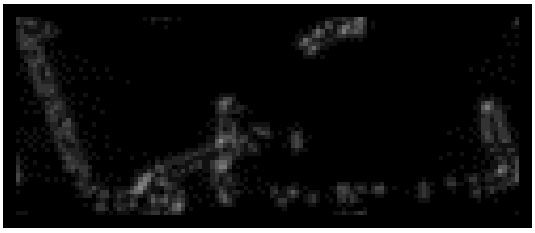
Table 3.6 : The best results achieved by each orientation

Filter Orientation	Recognition Rate	Codebook Size	Encoding
0º	65.98%	3072	KNN Assignment
30º	65.98%	3072	KNN Assignment
60º	67.87%	4096	KNN Assignment
90º	63.55%	4096	LS Assignment
120º	64.78%	4096	KNN Assignment
150º	67.50%	4096	LS Assignment



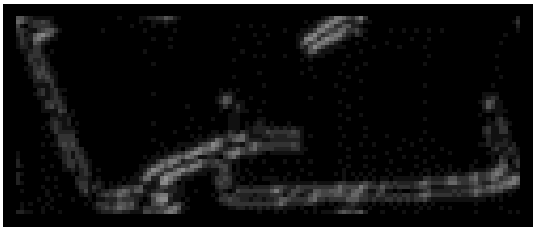
(a) Orientation 1 (0º)

The response on vertical text is strong but it is weak on the horizontal text



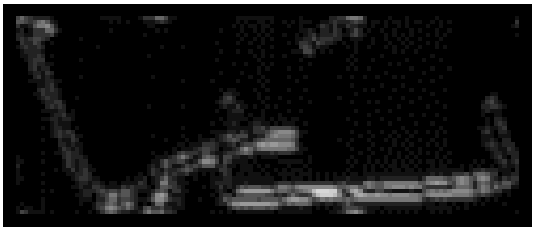
(b) Orientation 2 (30º)

The response on the vertical text became low and on the horizontal text became better



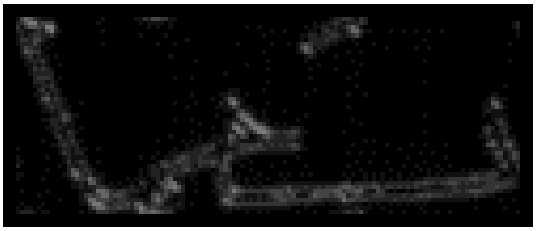
(c) Orientation 3 (60º)

The response on both horizontal and vertical text is observed



(d) Orientation 4 (90º)

The response on horizontal text is strong but it is weak on the vertical text



(e) Orientation 5 (120º)

The response on both horizontal and vertical text is observed

(f) Orientation 6 (150º)

The response on both horizontal and vertical text is observed

Figure 3.19: The response of Gabor filters with different orientations on a sample image

Experiments with Three Scales and Six Orientations

In the next Experiments, Gabor filter banks of three scales (wavelengths of 3, 6 and 12 pixels) and six orientations (0^o, 30^o, 60^o, 90^o, 120^o and 150^o) per scale are used to evaluate the effectiveness of increasing the wavelength of the Gabor filters. Therefore, we get 3×6=18 Statistical Gabor Features vectors -each of size 16 elements- for each image sample.

In the first set of experiments, the 16–D Statistical Gabor Features vectors corresponding to the individual orientation of each scale are applied to the BoF framework using codebook of size 4096 and the KNN Assignment encoding. Figure 3.20 shows the recognition accuracy achieved by the orientations of the different scales. The results showed that increasing the filters' scale negatively affected the recognition accuracies. This is due to the fact that large filters cover large text portions and hence the response became indiscriminative. Figure 3.21 visualizes the response of Gabor filters of the three scales on a sample image. The three filters have 60^o orientation. Concatenating the Statistical Gabor Feature vectors of the three scales per orientation significantly improved the recognition accuracies as the statistics of the multi scale filters produce scale-invariant representation. In all the cases, the best accuracies were achieved using the third orientation (60^o), followed by the sixth (150^o), first (0^o), second (30^o), fourth (90^o) and fifth (120^o). Highest recognition rate of 74.06% was obtained by concatenating the scales of the third orientation (60^o).

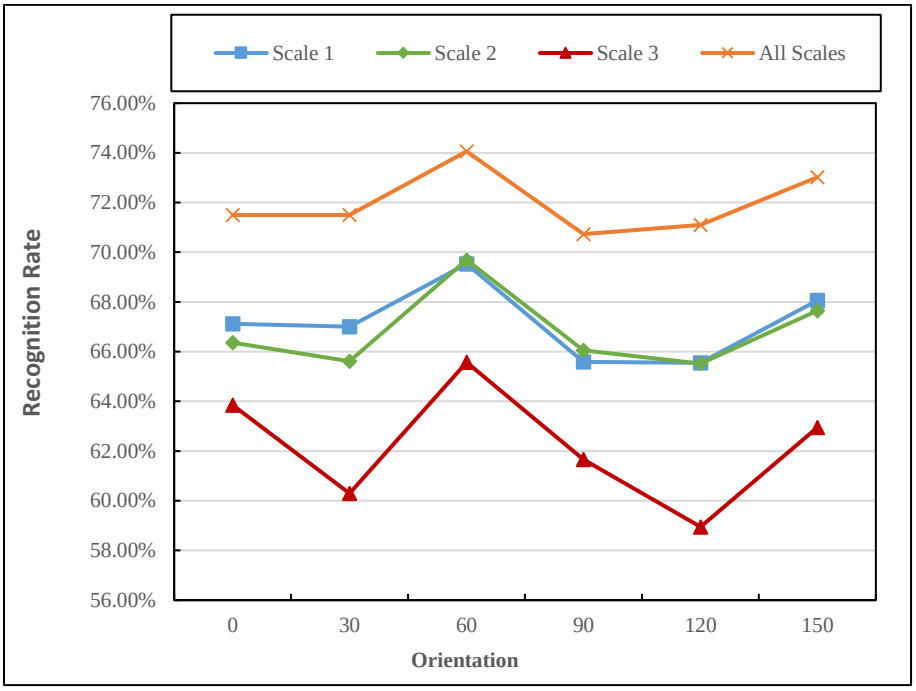
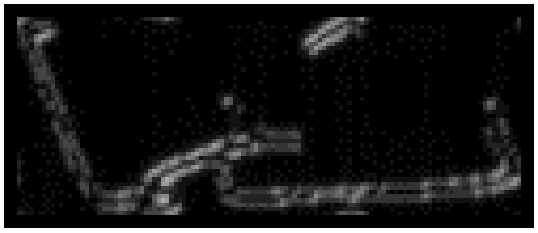
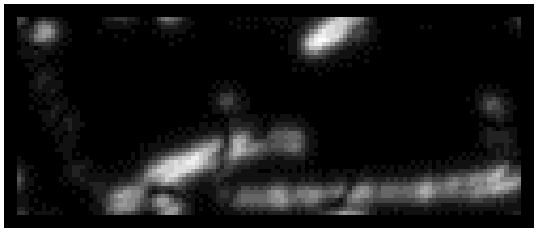


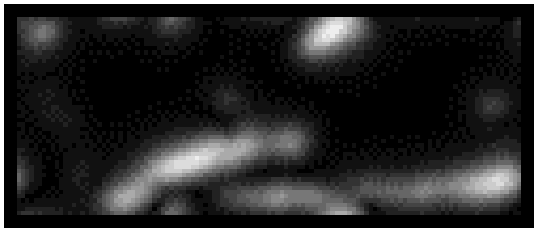
Figure 3.20: Recognition Rates for non-touching Arabic sub-word with Statistical Gabor Features of individual orientations of three scales



(a) Scale 1 (3 pixels)



(b) Scale 2 (6 pixels)



(c) Scale 3 (12 pixels)

Figure 3.21: The response of Gabor filters of the three scales on a sample image.

The three filters have 60o orientation

We also concatenated the Statistical Gabor Features of the orientations of each scale. The 16–D feature vectors corresponding to the orientations of each scale are concatenated. We start by concatenating the two orientations that achieved the best accuracies (the third (60^o) and the sixth (150^o)) and gradually add the first (0^o), the second (30^o), the fourth (90^o) and the fifth (120^o). Figure 3.22 shows the achieved recognition rates. We observed that concatenating the orientations performed better than the concatenation of the scales per orientation shown in Figure 3.20. Further, we observed that concatenating the orientations of the largest scale performed better. Best recognition result of 83.74% was achieved by concatenating the six orientations of the largest scale we used.

In the last set of experiments, we concatenated the Statistical Gabor Feature vectors of the three scales and the six orientations into a single vector of 288 (16×3×6) elements. The resulted features achieved recognition rate of 85.08% using codebook size of 4096 and the KNN Assignment encoding. This indicates that using the information of the whole scales and orientations significantly improved the discrimination of the BoF representation. This conclusion is inline with the results shown in the previous studies that applied the traditional statistical Gabor features in handwriting recognition (Mahmoud 2008) (Mahmoud & Al-Khatib 2010).

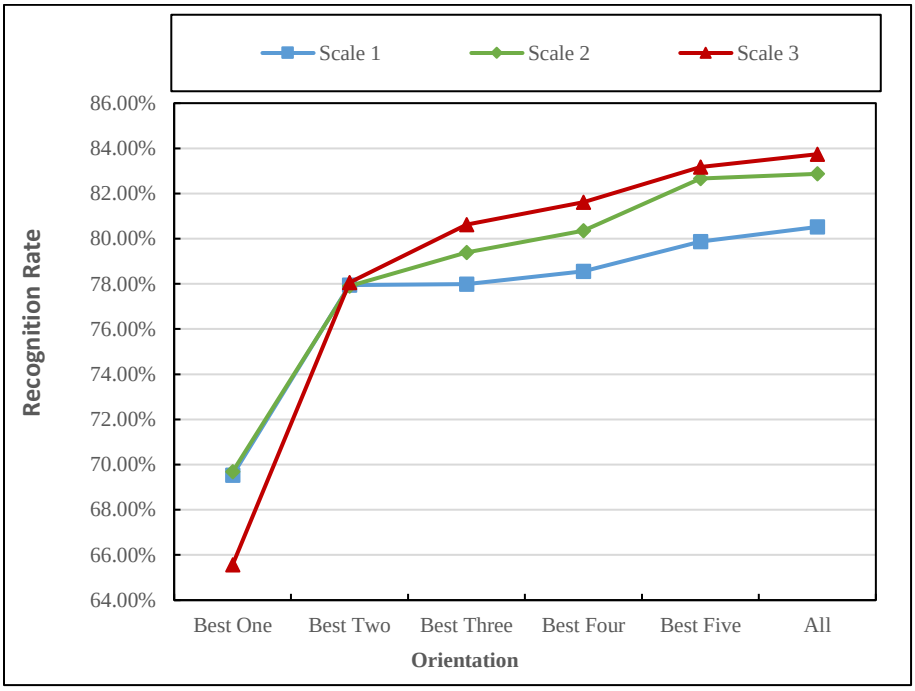


Figure 3.22: Recognition Rates for non-touching Arabic sub-word with the concatenation of the Statistical Gabor Features of the orientations of each scale

3.3.3.2. Evaluating Gabor Descriptors

Gabor descriptors combine the responses of all scales and orientations of the Gabor filter bank into a feature vector similar to the one produced by common local descriptors like SIFT and SURF.

In this work, we evaluated four pooling operations: Max, Sum, Mean and Variance. We also evaluated the normalization of the final vector to unit length, the approach applied by SIFT and SURF. In all experiments, we use Gabor filter bank of three scales (wavelengths of 3, 6 and 12) and six orientations (0^o, 30^o, 60^o, 90^o, 120^o and 150^o) per scale. Therefore, we obtain a vector of 72 (2×2×3×6) elements for each sample. These features were applied to the BoF framework using codebook of size 4096 and the KNN Assignment encoding.

In the first set of experiments, the performance of the four aggregation operations and the normalization was evaluated on samples of size 8×8 pixels. Figure 3.23 shows the Recognition accuracies. The recognition accuracies of Max, Sum and Mean operations were comparable while the Variance was worse. The normalization significantly improved the performance of all aggregation operations. Its impact in the Variance was recognized. Best recognition rate of 79.49 % was achieved by the Sum and Mean with normalization.

In the next set of experiments, we evaluated the effect of increasing the sample size. We draw samples of four scales: 8×8 , 12×12 , 16×16 and 32×32. Figure 3.24 shows the recognition rates of the four aggregation operations on the normalized Gabor Descriptors of individual scale. The results show that increasing the sample size improved the results of all aggregation operations. Highest recognition rate of 85.95% was achieved by the Sum and Mean operations.

In the last experiment, we used the normalized Gabor Descriptors of the three scales together. This combination achieves recognition rate of 86.44% using the Sum and Mean operations and the rate of 85.21% using the Max operation. This configuration is similar to the one used for extracting SIFT descriptors in Section 3.1.2. The results are still lower than those achieved using SIFT. Despite that, the Gabor descriptor forma enabled us to improve the performance of Gabor filter response features by applying techniques that proposed to enhance local descriptors, e.g., patch sub-regions and normalization.

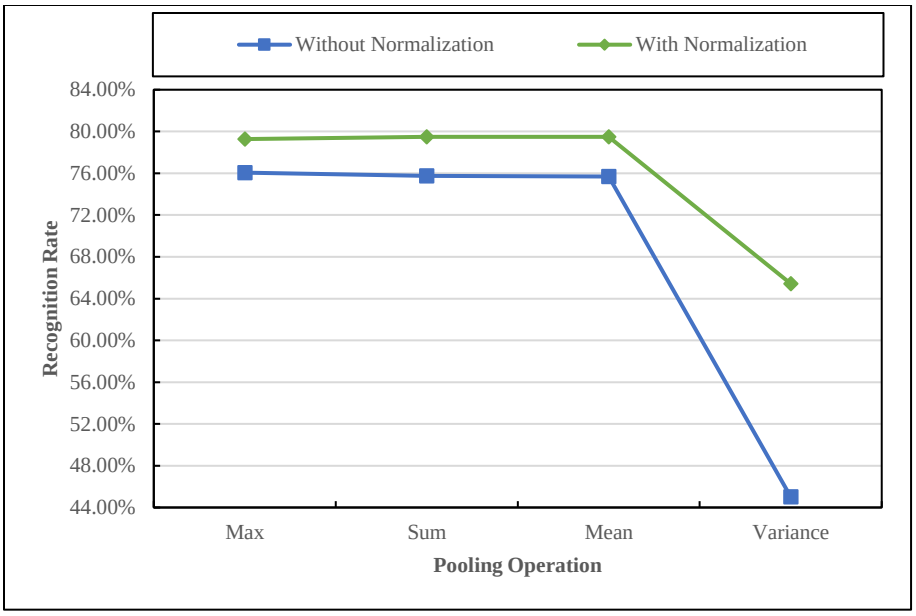


Figure 3.23: Recognition Rates for non-touching Arabic sub-word with different Aggregation Operations applied to Gabor Descriptors

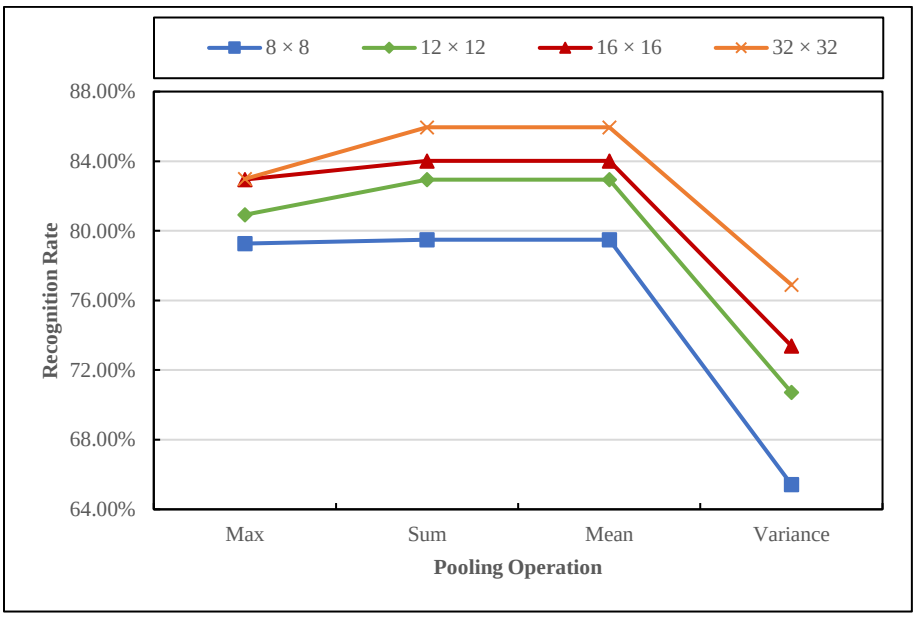


Figure 3.24: Recognition Rates for on non-touching Arabic sub-word with different Sampling Scales

3.4. Conclusions

This chapter presented a feature-learning framework for Arabic handwritten text recognition based on Bag-of-Feature (BoF). Several alternatives involved in the framework are investigated, and their effects in the learned features are evaluated. We used the Harris detector, Harris-Laplace detector and dense sampling. The selected regions are described using SIFT that produces 128-d descriptors for each region. The codebook is built by applying k-means clustering on sample vectors selected randomly from the training set. The feature vector of a given sample is obtained by quantizing its reduced SIFT descriptors using the codebook and constructing the normalized histogram of size equal to the codebook size. Two quantization approaches are evaluated, viz. hard and soft assignment. SVM is used in the classification phase. The system is evaluated on the non-touching Arabic Indian digits and Arabic subwords datasets of CENPARMI database.

The extensive experiments show that dense sampling outperforms the interest point detectors for handwritten text recognition. Codebooks with large sizes had a positive effect on the performance in all datasets. The best accuracies are achieved using a codebook of size 2048. Soft assignment reduces the side effect of quantization distortion associated with the hard assignment and improves the performance of the learned features. The developed framework achieved 99.34% recognition accuracy on the non-touching Arabic/Indian digit dataset, 95.53% recognition accuracy on the 10 most frequent classes of the non-touching Arabic subwords dataset and 89.93% on the full non-touching Arabic subwords dataset. All these results outperformed the state-of-the-art published work on the same datasets. The main source of errors in both datasets is the challenging writing styles. The two datasets contain samples written with complicated styles that are difficult even for human

recognition without context. The non-touching subwords dataset lacks enough training samples for some classes. Classes containing less than 10 training samples achieved poor performance. The accuracy of the 10 most frequent classes of this dataset support this claim. Despite that, our system achieved 100% accuracy on 6 digit classes, and successfully tolerated common problems encountered in published works such as imperfect and disconnected samples. On the full subwords dataset, our system achieved accuracy above 90% in many classes containing several hundred test samples, though some samples in these classes have complicated writing styles.

Utilizing the characteristics of the handwritten text images resulted in two novel versions of the SIFT algorithm that are computationally efficient and produce descriptors of half the size of the original SIFT. The two versions have achieved comparable recognition performance to SIFT on the non-touching Arabic Indian digits and Arabic subwords datasets, yet they are computationally efficient.

The Gabor filters features were arranged into two formats, viz., the *Statistical Gabor Features* (SGF) and *Gabor Descriptors* (GD). The experiments show that GD achieved better performance and allowed us to improve their efficiency by applying techniques that have been tested on local descriptors like normalization. Best recognition rate achieved on the non-touching Arabic subwords dataset is 86.44%. Further improvements, such as using more scales and orientations and applying several weighting approaches to the descriptor elements could enhance the results.

CHAPTER 4

FEATURE LEARNING FRAMEWORK FOR ARABIC HANDWRITTEN TEXT RECOGNITION

This chapter presents our proposed approaches for adapting BoF framework to HMM-based handwriting recognition systems by utilizing the characteristics of Arabic handwritten text to impose spatial localization in BoF representation. Section 4.1 presents how features are defined in HMM-based handwriting recognition systems and limitations of BoF framework on producing powerful features. Section 4.2 presents the approaches we propose for adapting the framework to HMM-based systems and to the characteristics of Arabic handwritten text. Section 4.3 presents the results of extensive experimentations on KHATT database and Section 4.4 concludes the chapter.

4.1. Feature Extraction in HMM-based Handwritten Text Recognition

Handwriting Recognition systems based on the Hidden Markov Models (HMMs) are the most successful systems for cursive text recognition of several scripts, including Arabic (Parvez & Mahmoud 2013) (Fink 2014). HMMs require sequence of observations representing the input patterns. The common approach for generating the sequence of observations for a text line image is the sliding window technique. The text line is scanned, along the text direction, by a window of few pixels width. For each window position, an observation vector is produced by extracting statistical features of the region under the

window. In order to propagate contextual relations along the sequence, the windows are overlapped. To impose localization on the window observation, the window is partitioned into vertical cells –which might be overlapped- where features are extracted from the cells, and then concatenated to form the window observation.

Our concern in this chapter is to utilize BoF framework to produce robust features for HMM-based Arabic handwritten text recognition. As the HMM-based handwritten text recognition systems require representations for narrow windows of the text image, the approaches of extracting local feature used by the normal BoF framework seem unsuitable. Interest region detectors might be unable to detect salient regions within the window and therefore no local features would be generated (Rothacker et al. 2012). Dense sampling approaches that are frequently applied to document image analysis applications (Rothacker & Fink 2015) (Aldavert et al. 2015) (Zagoris et al. 2014) require careful adaptation to ensure that the produced local features are true representation for the window contents. Moreover, local descriptor algorithms partition the spatial area of salient regions and dense samples into sub-regions for which statistical histograms are computed. Unless the small size of the salient regions and dense samples is respected, the histograms would be poor and the local descriptors become indiscriminate. The common shortcoming in the BoF representation is the lack of spatial information. Though the sliding windows impose spatial localization along the line direction, it is crucial to impose spatial localization within the window representation itself, as the location of text diacritics changes the meaning of the Arabic text, e.g., the two Arabic letters TAA (ـتـ) and YAA (ـيـ) have the same shape, yet they differ in the location of dots.

In this chapter, we adapt the local features extraction stage of the Bag-of-Features framework to the nature of the sliding window strategy. Then, we utilize the characteristics of the Arabic handwritten text for imposing spatial localization in the BoF representation. This adaptation motivated us to utilize multi-stream HMMs in a novel style that significantly improved the performance. The adaptations are presented in the next section.

4.2. Adapting BoF Framework to HMM-based Arabic Handwritten Text Recognition

In this chapter, the observations are a sequence of BoF representation. To generate a sequence of observations for a text line image, we follow the sliding window strategy which produces a sequence of narrow windows from the text image. Each window is partitioned into square cells of size $w \times w$ pixels, where w is the window width. The windows as well as the cells within the window might be overlapped. For each cell, one or more local descriptors are generated such that their centers coincide with the cells' centers. The descriptors are used to compute a global BoF representation for the window. Furthermore, for imposing spatial localization in the window representation, the window is partitioned into three regions based on the text baseline. The local descriptors whose centers are within the region are used in computing an independent BoF representation for that region. These independent representations motivate us to utilize multi-stream HMMs.

In the following subsections, we present the details of the adaptation of the local features extraction stage and the utilization of the writing baseline property of the Arabic handwriting for imposing spatial localization in the BoF representation. Then we discuss the utilization of the multi-stream HMMs in modeling the BoF representation of the three regions.

4.2.1. Local Features Extraction and Representation

For each of the window cells, several descriptors of different spatial scales are computed. The descriptors of the first scale have size identical to the cell size. The descriptors of the second scale expand the cell boundaries by 2 pixels from each side, the third scale by 4 pixels from each side and the forth scale by 6 pixels from each side and so on. This strategy ensures that all the descriptors are aligned with the cell's center. For cells of 8×8 pixel size, the 4-scale descriptors have sizes of 8×8, 12×12, 16×16 and 20×20. Figure 4.1 visualizes the 4-scale descriptors on a sample text.

Computing SIFT descriptors for overlapped cells is equivalent to the dense sampling strategy applied frequently in computing BoF representations (Nowak et al. 2006). The extraction of multi-scale descriptors in dense sampling is a common practice (Bosch et al. 2007) (Chatfield et al. 2011) (Avila et al. 2013) (Rusiñol et al. 2011) (Aldavert et al. 2015), as the multi-scale descriptors provide scale invariance. Aligning the multi-scale descriptors to the cell center enables them to represent the cell neighborhood at different scales. This procedure ensures that each window produces a fixed number of local features, in contrast with the interest regions that might not be detected in some windows. Furthermore, the large number of the produced local descriptors enriches the BoF representation. The number of the extracted local features depends on the image height, the window width and the cell's stride parameter that determines the degree of cells' overlapping (the smaller the stride, the more the overlapping) and the number of the descriptor's scales. Table 4.1 shows the number of the extracted local features for different windows, cell strides and descriptor scales. The image height is normalized to 96 pixels.

Since the samples' spatial area is usually very tight, e.g., 8×8 pixels, the descriptor layout is modified. The spatial area is partitioned into 2×2 sub-regions, instead of the 4×4 sub-regions used in the normal SIFT algorithm. For each region, the 8-bins gradient magnitude and orientation histogram is computed. This gives a 32-D (2×2×8) descriptor for each cell. The special case is the narrowest window we use, the window of 4 pixels width, where the 4×4-pixel cells are considered as a single region. For such cells, the SIFT algorithm gives 8-D descriptors. Figure 4.2 compares the layout of the original SIFT to the layout we propose in this work.

The modification in the descriptor layout is crucial for the quality of the local descriptors. The 4×4 spatial regions used in the SIFT algorithm would produce poor gradient histograms due to the few pixels in the regions. For instance, partitioning the 8×8 cell into 4×4 regions gives 16 regions, each includes 4 pixels (2×2). Distributing the gradient magnitude of 4 pixels between the 8 orientation bins leads to poor histograms. In contrast, the 2×2 regions gives 4 regions, each of 16 pixels (4×4). The distribution of the gradient magnitude of 16 pixels between 8 orientation bins is more representative compared to the previous case.

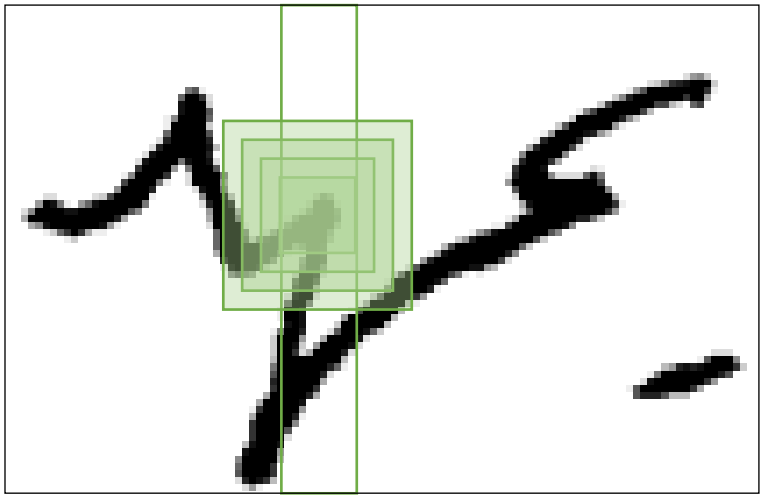
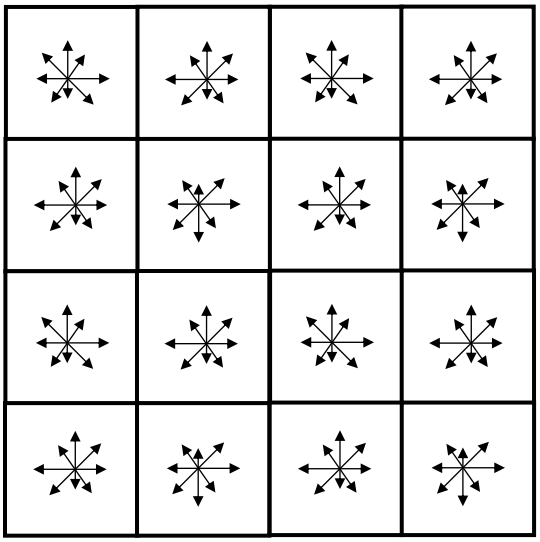


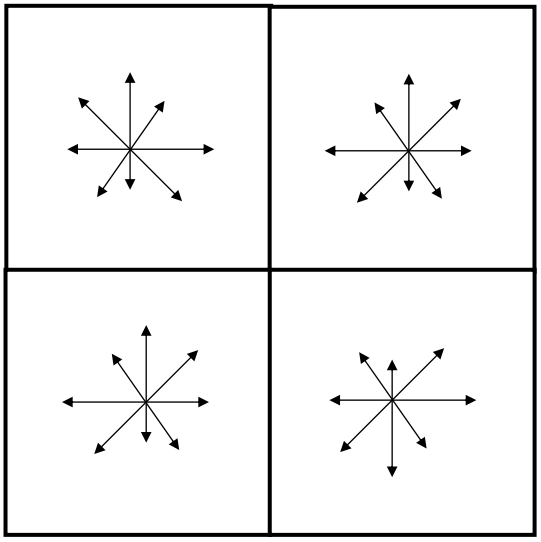
Figure 4.1: The 4-scale descriptors for 8-pixel window

Table 4.1: Number of the extracted SIFT descriptors for different values of windows width, cell strides and descriptor scales

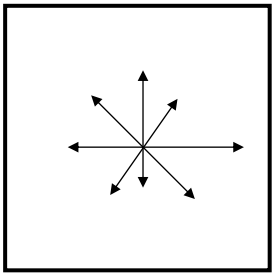
Window Width	Cell Stride	Descriptor scale	#Descriptors/Window
4	4	1	24
4	2	1	48
8	8	1	12
8	4	1	23
8	2	1	46
12	12	1	11
12	4	1	22
12	2	1	44
8	2	2	92
8	2	3	138
8	2	4	184



(a) Original SIFT.
The gradient histogram is computed in 4×4 sub-regions.



(b) Modified SIFT.
The gradient histogram is computed in 2×2 sub-regions.



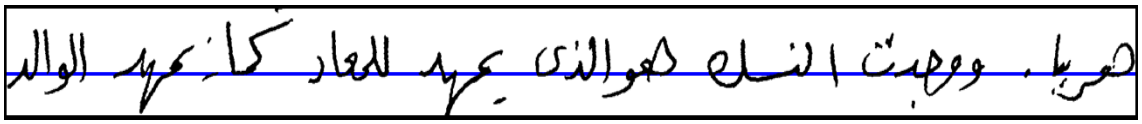
(c) SIFT for 4×4-pixel cells.
The gradient histogram is computed in a single region

Figure 4.2: The original SIFT layout vs. the modified SIFT layout

4.2.2. Imposing Spatial Localization in the BoF Representation

One of the shortcoming of the BoF representation is the lack of spatial localization in the representation, as the representation is a global histogram of the occurrences of the local features. To observe the spatial localization within the window representation, the window is partitioned into three vertical regions, utilizing the writing baseline property of Arabic text (Ahmad et al. 2016). The writing baseline of a text line is estimated based on the horizontal projection profile. The line image is partitioned into three regions such that the writing baseline is located in the middle region. The height of the middle region is adjusted such that it contains the main text. The lower and upper regions would contain the peripheral parts like the lower/upper parts of long letters and the diacritics as shown in Figure 4.3. Due to the large variability in the unconstrained handwritten text, the partitioning might not be accurate. However, it is useful for estimating the location of crucial text components like the dots. The local descriptors whose centers are within the region are used in computing an independent BoF representation for that region. The three BoF representations are concatenated to comprise the final window observation or they are modeled independently using multi-stream HMMs. Additionally, the global BoF representation could also be computed from the whole window descriptors and used to provide global window representation.

Our approach of partitioning the text image into sub-regions is similar to the Spatial Pyramid Matching (SPM) (Lazebnik et al. 2006), the successful BoF extension for observing spatial information in the representation. While the SPM partitions the image arbitrary, we utilize the characteristics of the Arabic handwritten text to partition the text line image into three meaningful regions.



(a) The estimated Baseline

(b) The text line is partitioned into three regions such that the middle region includes 50% of the black pixels

Figure 4.3: The estimated baseline and the boundaries of the three regions for a sample image from KHATT database

4.2.3. Integrating the Representation with the HMM system

Due to the large dimensionality of the BoF representation, it is hard to model them directly by the traditional continuous Gaussian-based Hidden Markov Models (GHMM). In (Rothacker et al. 2012), two approaches were proposed to deal with large dimensionality of the BoF representation. In the first approach, the BoF representations are reduced to lower dimensional vectors by applying the PCA. In the second, the BoF representation is interpreted as a probability distribution for the visual codewords and the HMMs are trained to estimate their weights only. In this work, we utilize the Discrete Hidden Markov Models (DHMM) to model the window representation. The DHMMs have been utilized before for the recognition of machine-printed text (Khorsheed 2007) (Al-Muhtaseb et al. 2008) (Ahmad et al. 2012) as well as for offline handwritten text (Dehghan et al. 2001) (Cheriet et al. 2007) (Alkhateeb et al. 2011) (Elzobi et al. 2013) (Mahmoud et al. 2014) (Jayech et al. 2016).

The independent BoF representations obtained from the partitioning of the text image into three regions motivate us to exploit the multi-stream DHMMs to model them. Though the multi-stream HMMs were utilized in developing offline handwriting recognition systems (Kessentini et al. 2010) (Ahmad et al. 2014) (Jayech et al. 2016), it is worthy to notice that these systems assume that the window observation is coming from independent feature streams, where each stream produces features for the entire window. The features of each stream are modeled independently in the HMMs. In our case, however, each region is treated independently, i.e., the BoF representation of each region is modeled independently by the HMMs. We assume that the middle region would provide representation for the

letter core shape while the upper and upper would provide information to discriminate the letters that have similar shapes e.g., the two Arabic letters TAA (ـتـ) and YAA (ـيـ).

4.3. Experimental Results

In this section we present the results of the extensive experimental evaluation of the proposed enhancements of the BoF framework. We start by describing the dataset we used in the experiments and our implementation of the recognition system. The evaluation of the impact of the various parameters using the validation set is addressed in Section 4.1 and the experimentation results using the test set are discussed in Section 4.2.

4.3.1. The Distinct Lines of KHATT Database

The experimentations were carried on the distinct lines of KHATT database, the public open-vocabulary database (Mahmoud et al. 2014). The dataset comprises 6712 distinct Arabic handwritten lines from large corpus written by 1000 writers of different ages, gender, educational level and handedness from different countries. The lines are distributed among training, validation and test sets. Table 4.2 shows useful statistics of the dataset.

Table 4.2: Statistics of the distinct handwritten lines in KHATT database

	#Lines	**#Words**	**#characters (with space)**	**#characters (without space)**
Training	4808	55,893	301,924	260,455
Validation	938	11,113	59,507	49,633
Testing	966	10,675	58,463	48,754
Total	**6,712**	**77,681**	**419,894**	**358,842**

The dataset was used to evaluate a HMM-based recognition system in (Mahmoud et al. 2014). Best character accuracies of 46.70% and 46.13% were reported on the validation and test sets, respectively. We found two other works experimented with KHATT database (Hamdani et al. 2013) (Stahlberg & Vogel 2015). However, both works used the distinct and fixed lines datasets in their experiments. Furthermore, the two works utilized a word-level language model and sophisticated sub-lexical approach to address out-of-vocabulary problem. The recognition accuracies were reported at the word level. These differences make these two works incomparable to our work. Indeed, we compare our results with those reported in (Mahmoud et al. 2014), since we are using the same HMM recognizer and the same dataset. The differences are in the feature extraction, the proposed adaptation techniques and the utilization of character-level bi-gram language model.

4.3.2. Handwritten Text Recognition

Our handwritten text recognition system takes a text line image and reports the transcription of the text. The system has three main phases, preprocessing, feature extraction and recognition. In the preprocessing phase, the line images are normalized to 96-pixel height while preserving the aspect ratio. Then the slant and skew of the text lines are corrected as in (Mahmoud et al. 2014). To observe the right-to-left writing style of Arabic language, the images are flipped left-to-right. Two blank windows are appended to the text lines, one at the beginning and one at the end, to ensure that the extracted observations cover the entire text in the line image. For extracting multi-scale descriptors, few white pixels might be padded to the boundaries of the line image to ensure that all descriptors are aligned with the cells' centers.

In the feature extraction phase, the BoF representation is computed. SIFT descriptors are computed for the cells of the sliding window. The zero-valued descriptors are excluded from the codebook learning and quantization steps as in (Law et al. 2014). The non-zero descriptors are de-correlated by applying PCA and quantized to the closest codeword in the codebook. The codebook is learned by applying the k-mean clustering algorithm on a set of one million de-correlated descriptors selected randomly from the training samples. The codebook size is tuned in the validation set and the best value is used in the test set evaluation. The final window representation is the BoF representation obtained by the average pooling. We utilized the implementations provided by the VLFeat open-source library (Vedaldi & Fulkerson 2010) for implementing the different algorithms of the BoF framework.

In the recognition phase, we utilize the discrete HMMs to cope with the large dimensionality of the observation vectors. The observation vectors are quantized into discrete symbols based on a predefined codebook that is learned from the training samples. The different shapes of the Arabic characters are modeled by separate Markov models. The total number of models in the system is 153, corresponding to all character shapes, ligatures, digits and punctuation marks in the database. In the final results, the different shapes of the same characters are merged and considered as a single model. The total number of these shapeless classes is 61. The number of states in the models are tuned in the validation set and the best value is used in the test set evaluation. In this evaluation, all models have the same number of states using Bakis topology as the models with variable number of states didn’t show significant improvements in the initial experiments. The models are trained based on the two-phase training strategy. In the first phase, the

parameters of a single model are initially estimated from the training observation sequences. In the second phase, the initialized model is cloned to all the models in the system and they are re-trained by several iterations of Baum-Welch algorithm. The recognition is performed by the Viterbi decoding with character bi-gram language model that learned from the transcriptions of the training set. We utilized the HTK toolkit (Young et al. 2006) for implementing the recognition system described above. Though the HTK toolkit has auxiliary tools for k-mean clustering (HQUANT) and vector quantization (HCOPY), we use the VLFeat implementations for k-means clustering and vector quantization.

4.3.3. Parameter Evaluation using the Validation Set

We conducted several experiments on the validation set in order to study the impact of the various parameters on the system performance. We start by studying the impact of the sliding window parameters and the local features.

A sliding window of few pixels width is used to generate a sequence of observations for a line image. The window is partitioned into vertical cells for which the SIFT descriptors are computed. The windows as well as the cells within the window might be overlapped. The window stride determines the windows overlapping while the cell stride determines the cells overlapping. The window width and stride, the cell stride and the number of the extracted descriptors for a cell have high impact on the system performance. The larger the window is, the more the contextual information are observed. However, wider windows might cover more than a single character. Smaller window stride helps in generating sufficient observations for the line image and smaller cell stride helps in generating more descriptors. Multi-scale descriptors capture more contextual information and provide scale

invariance. They also increase the number of the local features. Table 4.3 shows the system performance on the validation set using different window widths, window strides, cell strides and the multi-scale descriptors. The performance is given in the character accuracy rate where the substitution, deletion and insertion errors are discounted. In all experiments, the size of the BoF and HMM codebooks are 256, as this value achieved the best performance in the initial results. Later, we try several values for the final system (See Table 4.5). The number of the HMMs states are varying between 4 and 14 states and the best result is reported. The experiments showed that best accuracies were achieved using the number of states in this range (Table 4.6).

Table 4.3: The impact of window width, window stride, cell stride and number of descriptor's scales on the validation set

Window Width	Window Stride	Cell Stride	#Scales	Character Accuracy %
4	4	4	1	37.10%
4	4	2	1	39.80%
4	2	2	1	41.30 %
8	8	2	1	38.10%
8	4	2	1	43.30%
8	2	2	1	42.30%
12	6	2	1	43.20%
8	4	2	2	45.90%
8	4	2	3	47.10%
8	4	2	4	48.40%

The results show that windows of 8-pixel width have better performance than the 4- and 12-pixel width windows. Half overlapping of the 4- and 8-pixel width windows gave best results. Overlapping of 75% of the 8-pixel width window (i.e., the window stride of 2) doesn't improve the results. The 2-pixel cell stride achieved better performance than the 4-pixel stride in the 4-pixel width windows. The multi-scale descriptors have high impact on the accuracy. The 4-scale descriptors with the 8-pixel width windows improved the performance by 5.10% accuracy (it achieved 48.40% accuracy, compared with 43.30% accuracy of the single-scale descriptors). In the following evaluations, we use the values that achieved the best performance in this section, i.e., windows of 8-pixel width with 4-piexel stride (half-overlapping) and 2-pixel cell stride. For each cell, the 4-scale SIFT descriptors are extracted.

To assess the impact of utilizing the writing baseline and the multi-stream HMMs, we setup three more configurations. In the first configuration, the BoF representations of the three regions were concatenated and fed to a single-stream HMMs-based system. In the second configuration, the BoF representations of the three regions are fed to 3-stream HMMs-based system, and in the third configuration, a 4-stream HMMs-based system is used, where the forth stream represents the global BoF representation of the entire window (before partitioning into the three regions). In all configurations, the streams were weighted equally. The recognition accuracies of these systems are shown in Table 4.4. Utilizing the writing baseline has improved the performance by 4.90%, while the 3-stream HMMs system added extra 9.80% to the accuracy. The 4-stream HMMs system achieved 64.10% accuracy on the validation set, which is much better than the best state-of-the-art results

reported in (Mahmoud et al. 2014) on the same dataset (46.70%) using the intensity and gradient statistical features.

Table 4.4: The performance utilizing the writing baseline and the multi-stream HMMs on the validation set

Recognition System	Character Accuracy %
1-stream HMMs	53.30%
3-stream HMMs	63.10%
4-stream HMMs	64.10%

Other crucial parameters that we have examined are the size of the BoF and HMM codebooks and the number of models' states. Table 4.5 shows the performance of the system for different sizes of the BoF and HMM codebooks. The best performance is achieved by the codebooks of size 256. In Table 4.6 we show the performance for different number of states for the 256 codebooks. The best performance is achieved using the 10-states system.

The performance of the Viterbi decoding is sensitive to the *word insertion penalty* and *grammar scale factor parameters*. It is highly recommended to tune these parameters on the validation set (Young et al. 2006). In all of the above reported results, the parameters we used the default values for both parameters which are 0 and 1, respectively. The performance for different s and p values are shown in Table 4.7. The results show that carful tuning of the two parameters has high impact in the performance. The best accuracy achieved is 63.10%.

Table 4.5: The performance of using different sizes for BoF and HMM codebooks

BoF Codebook Size	HMM Codebook Size	Character Accuracy %
128	256	56.50%
256	128	56.20%
256	**256**	**59.10%**
256	384	57.70%
384	256	58.10%

Table 4.6: The performance of using different number of states

#States	Character Accuracy %
8	56.00%
10	**59.10%**
12	53.70%
14	51.80%

Table 4.7: The performance of tuning the grammar scale factor and word insertion penalty parameters of the Viterbi decoding

Grammar Scale Factor	Word Insertion Penalty	Character Accuracy %
-1	1	59.50%
-1	10	50.20%
0	1	59.10%
0	**3**	**63.10%**
0	10	51.30%
1	1	58.70
1	**4**	**63.10%**
1	10	52.60%
2	1	58.20%
2	**4**	**63.10%**
2	10	58.20%
3	1	57.70%
3	**4**	**63.10%**
3	10	54.20%

4.3.4. The performance on the Test Set

The recognition performance on the test set is presented in Table 4.8. The best configuration using the 4-stream HMMs system has achieved 63.40% accuracy on the test set. This result outperforms the best results in (Mahmoud et al. 2014) on the test set (46.13%). The results show the power of the BoF framework in learning robust representations for handwritten text. They indicate also that the carful adaptation of the framework to the HMM-based text recognition produces better observations for the handwritten text. Furthermore, exploiting the characteristics of the Arabic script in constructing the BoF representation has improved the quality of the produced observations.

The contribution of the representations of the different streams can be adjusted by assigning different weights to the streams. To assess the impact of the stream weighting, we conducted set of experiments on the 4-stream HMMs-based system where different stream weights were used. We found that assigning higher weights to both the global window and middle region representations improves the character accuracy rate. The new rates we achieved on the validation and test sets are 65.00% and 64.30%, respectively.

Table 4.8: The recognition accuracy on the Test set

Recognition System	Character Accuracy %
1-stream HMMs	52.50%
3-stream HMMs	61.10%
4-stream HMMs	63.40%

4.3.5. Comparison with Traditional Statistical Features

In this section, we compare the performance of the BoF representation with traditional statistical features on the multi-stream DHMMs-based system. The Intensity and Gradient Statistical Features (IGSF) that achieved the best performance in (Mahmoud et al. 2014) are extracted. A sliding window of 8-pixel width and 4-pixel stride was used. The window is partitioned into 8×8-pixel cells with 2-pixel stride. The IGSF represent a cell by three values corresponding to the summation of (1) the cell's pixel intensity, (2) the horizontal derivative and (3) the vertical derivative. The window is represented by 144-D observation vector, obtained by concatenating the cells' features. The IGSF were used with several discrete HMM-based recognition systems. The first system is a single-stream HMMs where they are used alone. The second system is a 2-stream HMMs where the sliding window is represented by two independent features, viz., the IGSF and the BoF representation for the single-scale SIFT descriptors. The third system is 4-stream HMMs where the IGSF are used as a global window representation with the three BoF representations of the window's regions. The recognition performance of the three systems on the validation and test sets is shown in Table 4.9.

The performance of the single-stream HMMs using IGSF is slightly below that reported in (Mahmoud et al. 2014), since we have used wider window than their window that achieved the optimal results. The 2-stream HMMs system has achieved accuracy of 55.30% on the test set, which is much better than that achieved by the single-stream HMMs using individual features. The 4-stream HMMs with IGSF and the three BoF representations has achieved comparable performance to the 4-stream HMMs system with BoF representation as global window representation shown in Table 4.8.

Table 4.9: The performance BoF representation and traditional statistical features on the validation and test set

Recognition System	Character Accuracy %	
	Validation Set	Test Set
1-stream HMMs using IGSF	43.10%	42.70%
2-stream HMMs using IGSF + BoF	55.80%	55.30%
4-stream HMMs using IGSF + BoF	64.00%	63.40%

4.4. Conclusions

In this chapter, we adapt the Bag-of-Features representation to the nature of the sliding window of HMM-based systems. The local descriptors are extracted densely at multiple scales that are aligned to the cells' centers. The layout of the SIFT descriptors is modified to be commensurate with the spatial size of the small cells. To impose localization in the representation, the writing baseline of the Arabic text is utilized in partitioning the window into three regions where each region is represented by an independent BoF representation. The region representation are modeled using multi-stream discrete HMMs. The extensive experimentations illustrate that our strategy in extracting multi-scale local descriptors and aligning them to the center of the windows has high impact on the quality of the produced representation. Partitioning the text line into three regions based on the writing baseline helps in adding localization to the BoF representation. The utilization of the multi-stream DHMMs enhances the performance of the recognition system. The best system has achieved accuracy rate of 64.30% on KHATT database using 4-stream HMMs. The results are promising as the dataset is challenging with unconstrained natural handwritten text.

CHAPTER 5

BERNOULLI HIDDEN MARKOV MODELS FOR ARABIC HANDWRITTEN TEXT RECOGNITION

In this chapter we present our implementation of the segmentation-free handwritten text recognition system based on Bernoulli Hidden Markov Models (BHMMs). Then, we present the two approaches we proposed for enhancing the quality of the binary observations produced by the sliding window strategy. Section 5.1 provides an overview of BHMMs and reviews their application to handwriting recognition. Section 5.2 presents the two proposed approaches. Section 5.3 shows the experimentation results and Section 5.4 concludes the work.

5.1. Bernoulli Hidden Markov Models

A Bernoulli Hidden Markov Model (BHMM) is a hidden Markov model in which the state emission probability is modeled by multivariate Bernoulli mixtures. Assuming $qt = j$ be the current state at time t, the probability that the system would generate the binary observation ot (denoted by $b_j(o_t)$) is given by

$$b_j(o_t) = \sum_{k=1}^{K} \pi_{jk} \prod_{d=1}^{D} \left(p_{jkd}\right)^{o_{td}} \left(1 - p_{jkd}\right)^{1-o_{td}}$$

Where K is the number of the mixture components, π_{jk} is the prior probability of the k^{th} mixture of state j, D is the dimension of the binary observation o_t, p_{jkd} is the probability that the d^{th} bit in the binary observation o_t would be 1 according to the k^{th} mixture of state j. Finally, o_{td} is the d^{th} bit in the binary observation o_t (Giménez & Juan 2009a).

The BHMM-based handwritten text recognition system was first proposed in (Giménez & Juan 2009a). The state emission probability was modeled by a single multivariate Bernoulli probability density function. The text images were scaled to 30 pixels height while maintaining the aspect ratio and then converted to binary images using Otsu threshold method (Otsu 1979). The columns of the binary images are taken as the observations. The system was evaluated on isolated English words extracted from IAM database (Marti & Bunke 2002). The character recognition error rate of 44.00% was reported by using BHMMs of 10 states. For the sake of comparison, the same database was used to evaluate a traditional HMM-based handwriting recognition system with single multivariate Gaussian probability densities and real-valued observations. Character recognition error rate of 64.20% was reported by using HMMs of 8 states. In (Giménez & Juan 2009c), the single multivariate Bernoulli probability density was replaced by multivariate Bernoulli mixtures. This improvement dropped down the error rate on the above dataset from 44.00% to 30.90% when 64-mixture states was used. In (Giménez & Juan 2009b) the system was evaluated on a more challenging dataset comprising English text lines extracted from IAM database. Best recognition error rate of 42.10% was achieved by using of 6-state models and 64 mixtures per state. To capture contextual information in the observations, the sliding window technique was proposed in (Giménez et al. 2010). A narrow sliding window of few columns is passed over the text line with a stride of one pixel. The columns under the

window are concatenated and taken as a single observation. The impact of the sliding window technique was assessed on Arabic handwritten text using IfN/ENIT database (Pechwitz et al. 2002). Character recognition error rate of 12.30% was achieved by a sliding window of 9 pixels. To reduce the effect of image distortion, the sliding window repositioning technique was proposed in (Alkhoury et al. 2012). The sliding window is translated such that the window center is aligned with the center of mass of the text portion overlaid by the window. The observation is constructed from the columns overlaid by the translated window. The BHMM-based recognition system with the sliding window and sliding window repositioning techniques won the ICFHR 2010–Arabic Handwriting Recognition Competition (Margner & El-Abed 2010). To assess the impact of the sliding window repositioning technique in reducing the vertical image distortion, it was applied in (Doetsch et al. 2012) to the traditional Gaussian-based HMMs recognition system. The system was compared with the Long-Short-Term-Memory (LSTM) which is powerful in tolerating the vertical image distortion. The experiments carried on Arabic IfN/ENIT and French RIMES datasets (Augustin et al. 2006) showed that window translation improves the recognition accuracies of both the HMM- and LSTM-based systems.

Despite the prominent recognition performance that was reported on IfN/ENIT database, the IfN/ENIT database is of limited-vocabulary comprising isolated words of Tunisian villages and cites names, so it lacks the naturalness of handwritten Arabic text. To nominate the system for recognizing natural unconstrained Arabic handwritten text, it must be subjecteed to extensive assessment using comprehensive dataset that gives true representation of Arabic handwritten text. In this chapter, we evaluate the BHMM-based handwritten text recognition system on KHATT database which is a public open-

vocabulary database of unconstrained Arabic handwritten text. It contains 6712 distinct handwritten lines from large corpus written by 1000 writers of different ages, gender, educational level and handedness from different countries (Mahmoud et al. 2014). The purpose of this evaluation is to investigate how BHMM-based system perform on such a challenge dataset and to come up with the best configurations of the observation dimensionality, the number of states per models, the number of Bernoulli mixtures per state and the sliding window width. As the prominent performance is usually attained by using a wider sliding window, we propose two approaches that would reduce the dimensionality of the binary observations produced by the sliding window technique. Besides the dimensionality reduction, the second proposed approach would impose spatial localization to the observations by partitioning the window into small vertical cells prior to observation construction. The two proposed approaches are presented in the next section.

5.2. Local Sampling and Local cell layers

The two proposed approaches are dealing with a window of $h \times w$ pixels where h is the image height and w is the window width. We assume that w is odd and the window stride is always one pixel. Therefore, the i^{th} window is centered at the i^{th} column to represent it with the context around.

5.2.1. Local Sampling

In the local sampling approach, the columns that are farther from the window center (in the left and right sides) are alternatively sampled (Figure 5.1). We argue that the farther columns would be represented well by their windows. For the current window, a partial view of them would be sufficient for capturing the contextual information. We empirically

choose to preserve at least one-third of the columns in the center of the window non-sampled. The left and right sides are reduced to half by sampling them alternatively. The observation is composed by serializing the sampled left columns, the middle and the sampled right columns. This approach reduces the observation dimensionality approximately by a third. For 32-pixel height images, the 9-pixel width window produces 288-D observations while the local sampling approach produces 192-D observations which are two-thirds the dimensionality.

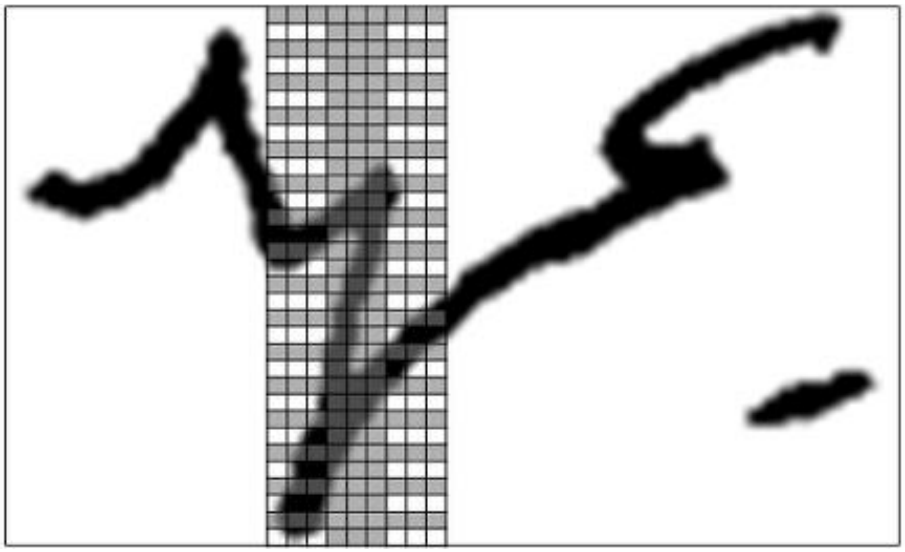


Figure 5.1: Local sampling for a window of size 32×9

The three columns from the left and right sides are alternatively sampled. The final window observation is the concatenation of the dark elements.

5.2.2. Local Cell Layers

The local cell layers approach partitions the window into non-overlapping square cells of size $w \times w$. The cells are processed in a layered fashion. Layer 1 consists of the 8-neighbors of the center pixel. Layer 2 consists of the 16 pixels surrounding the 8-neighbors and Layer 3 consists of the 24 pixels surrounding the 16 pixels in Layer 2 and so on (Figure 5.2 (a)). Eight pixels are selected from each layer, viz., the pixels lying on the horizontal, vertical and the two diagonal axis passing through the cell center as shown in Figure 5.2 (b). The selected 8 pixels are serialized, giving 8-D binary vector. The cell feature vector is the concatenation of the 8-D binary vectors of each layer. Correspondingly, the window observation is the concatenation of the cells' feature vectors.

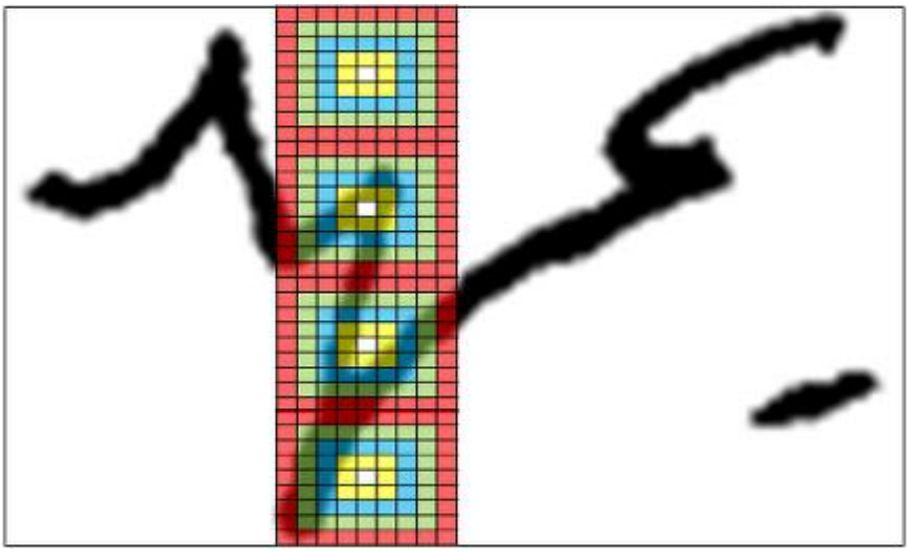
Partitioning the window into vertical cells is common practice in traditional HMM-based handwriting recognition systems in order to impose localization in the window observations. The selection of 8 pixels from each layer is inspired by the Local Binary Pattern (LBP) (Ojala et al. 1996) (Ojala et al. 2002) that are computed at several radius. Unlike the LBP that thresholds the gray-level intensity of the pixels lying on fixed distance from the center, we take the binary intensity values of the 8 pixels that (approximately) lie on fixed distance from the cell center.

The local cell layers approach has two advantages. The first imposes localization in the window observations due to cell partitioning and the second significantly reduces the observation dimensionality. For a sliding window of w pixels width, the local cell layers approach produces a binary observation of $\left\lceil \frac{h}{w} \right\rceil \times \left(\frac{w-1}{2} \right) \times 8$ bits. For wider windows, this

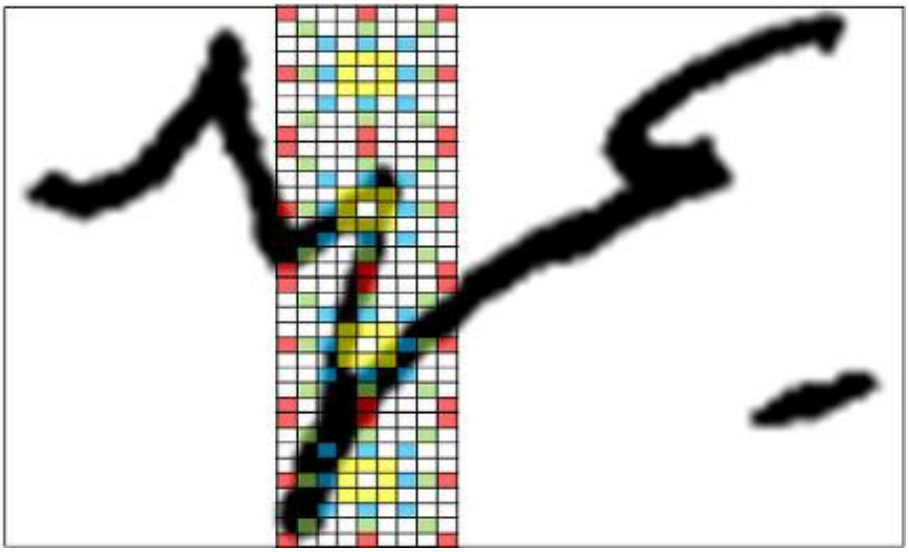
reduction becomes significant. Table 5.1 compares the dimensionality of the observations produced by the local cell layers with that of the original window for 32-pixel height.

Table 5.1: The Observation dimensionality of the original System vs. the local cell layers approach

Width	Original Dimensionality	Local cell layers Dimensionality	Reduction %
3	96	88	8.33%
5	160	112	30.00%
7	224	120	46.43%
9	288	128	55.56%



(a) The four Layers of 9×9 cells are presented in different colors



(b) The selected pixels from each layer are shown in the layer's color

Figure 5.2: Local cell layers for a window of size 32×9

5.3. Experimental Results

To assess the performance of the BHMM-based handwritten text recognition system on unconstrained Arabic handwritten text, we have conducted extensive experimentations on the distinct handwritten lines of KHATT database (Mahmoud et al. 2014) (See Table 4.2 for the useful statistics of the dataset). The line images are in gray-level format. The line slant and skew were corrected using the procedures presented in (Mahmoud et al. 2014). To acquire binary observations, the lines were binarized using Otsu threshold method and normalized to a fixed height while preserving the aspect ratio. Then, they were flipped left-to-right for maintaining the writing style of Arabic.

The BHMM recognition system is defined, trained and evaluated according to the procedure presented in (Giménez & Juan 2009b) and (Alkhoury et al. 2012). The system comprises as many BHMMs as the number of symbols in the dataset. The dataset in hand has 153 symbols corresponding to all character shapes, ligatures, digits and punctuation marks. Each BHMM has a fixed number of states with linear topology. The models states are first defined with a single Bernoulli mixture and then the number of mixtures are grownup as explained later. The model parameters are initialized by uniformly segmenting the training set and applying the Viterbi initialization. The initialized models are trained by running 8 iterations of the Expectation-Maximization procedure using the entire training set. To create BHMM with $K > 1$ Bernoulli mixtures, the mixtures of the trained models of $K/2$ mixtures are split. The created models are trained by running 4 Expectation-Maximization iterations on the entire training set. The trained system is evaluated on the validation set using the Viterbi algorithm. The configurations that achieved the best performance on the validation set are used in the final evaluation on the test set.

We utilized the transLectures-UPV toolkit (Del-Agua et al. 2014) for implementing the system. To report the performance, the obtained transcriptions are aligned with the ground truth transcriptions using the HRESULT command of the HTK toolkit (Young et al. 2006). Before the alignment, the different shapes of the same characters are merged and considered as a single class. The performance is reported at character recognition accuracy rate where the substitution, deletion and insertion errors are discounted.

5.3.1. The Baseline System

The first set of experiments was conducted to evaluate the impact of the image height, the number of states per model and the number of the mixture components per state. We evaluated five values for the image height (24, 32, 40, 48, 64), four values for the number of states per model (4, 6, 8, 10) and six values for the number of Bernoulli mixtures per state (2, 4, 8, 16, 32, 64) which resulted in 5×4×6 = 120 combinations. In all experiments, the window width is one pixel and therefore the observation dimensionality is as long as the image height. The character recognition accuracy rates achieved using these parameters are shown in Table 5.2.

The results showed that regardless of the observation dimensionality and the number of states, increasing the number of Bernoulli mixtures improves the character accuracy rate. This is attributed to the fact that the simple multivariate Bernoulli density can't model the dependency and correlation between the bits of binary images (Juan & Vidal 2004). Using large number of mixtures would help modeling them properly. We expect that increasing the number of Bernoulli mixtures beyond 64 mixtures per state would improve the recognition accuracy. However, due to the computational overhead associated with using the large number of Bernoulli mixtures we didn't extend our evaluation beyond 64 mixtures

per state. The best number of states is proportional to the observation dimensionality, the higher dimensional observations require Bernoulli models with more states to model the large variations of the higher dimensional space. However, increasing the number of the states beyond the optimal value dropped down the recognition accuracy. The best character recognition accuracy rates were achieved using 32-D observations and 6-state models, though the accuracies achieved by fewer Bernoulli mixtures (less than 16 mixtures) were outperformed by the counterparts of the 24-D observations and 4-state models. The best character accuracy rate (44.92%) was achieved using 32-D observations and 6-state models with 64 Bernoulli mixtures per state. In the following evaluations, we used the 32-pixel height images and 6-state BHMMs.

Table 5.2: The character recognition accuracy rates of the BHMM recognition system using different observation dimensionality, number of model states and number of mixtures per state

#States	# Mixtures					
	2	**4**	**8**	**16**	**32**	**64**
4	35.89%	38.14%	40.39%	42.05%	43.53%	**44.35%**
6	34.58%	36.32%	38.99%	40.60%	41.75%	42.56%
8	30.13%	31.75%	33.96%	34.69%	35.34%	35.20%
10	24.57%	25.73%	27.31%	28.45%	29.67%	29.61%

(a) 24-D Observations

#States	# Mixtures					
	2	**4**	**8**	**16**	**32**	**64**
4	33.95%	36.08%	38.23%	40.35%	42.03%	43.29%
6	35.33%	37.52%	40.06%	42.32%	43.92%	**44.92%**
8	33.49%	35.18%	38.15%	40.19%	41.79%	42.64%
10	29.94%	31.49%	34.09%	35.65%	36.41%	36.85%

(b) 32-D Observations

#States	# Mixtures					
	2	**4**	**8**	**16**	**32**	**64**
4	30.31%	32.65%	34.89%	36.77%	39.18%	40.65%
6	34.16%	36.62%	39.46%	41.48%	43.28%	**44.59%**
8	34.37%	36.21%	39.04%	41.25%	43.13%	44.15%
10	32.14%	34.24%	36.84%	39.22%	40.91%	41.92%

(c) 40-D Observations

#States	# Mixtures					
	2	**4**	**8**	**16**	**32**	**64**
4	26.14%	28.77%	31.00%	33.37%	35.69%	37.35%
6	32.58%	35.22%	37.75%	39.86%	41.74%	43.14%
8	33.98%	36.58%	38.97%	41.11%	43.14%	**44.27%**
10	33.32%	35.19%	38.24%	40.53%	42.42%	43.06%

(d) 48-D Observations

#States	# Mixtures					
	2	**4**	**8**	**16**	**32**	**64**
4	16.62%	19.25%	22.55%	25.39%	27.91%	29.69%
6	27.75%	30.38%	32.52%	35.18%	37.04%	38.50%
8	31.54%	34.51%	36.73%	39.33%	41.41%	42.43%
10	33.02%	35.76%	38.15%	40.68%	42.66%	**43.82%**

(e) 64-D Observations

5.3.2. Sliding Window and Sliding Window Repositioning

The second set of experiments was conducted to evaluate the sliding window and the sliding window repositioning techniques. We experimented with four values for the sliding window width (3, 5, 7 and 9 pixels) with a stride of one pixel in all cases. The sliding window technique produces observations of $h \times w$ bits. The character recognition accuracy rates for the different window widths are shown in Table 5.3. Comparing the results of the results of Table 5.2, we noticed that the character recognition rate have significantly improved with wider windows. The 9-pixel width window added 8.51% to the character recognition rate. The performance of applying the repositioning strategy on the sliding window of 9-pixel width is also shown in Table 5.3 with the asterisk (*). The strategy achieved character recognition rate of 59.44% which outperforms the baseline system by 14.52%.

It is well known that the recognition accuracy of HMMs-based systems could be enhanced by balancing the contribution of the language model and controlling the word insertion penalty (Young et al. 2006) (TransLectures-UPV-Team 2014). This is achieved by tuning the grammar-scale-factor and word-insertion-penalty parameters of the Viterbi implementation. In Table 5.4, we show the character recognition rates for different values for the grammar-scale-factor and word-insertion-penalty using the 9-pixel windows and repositioning with 6-state BHMMs and 64 mixtures. The character recognition rate on the validation set increased to 63.41%.

Using the best configurations (32-pixel image height, 9-pixel window with reposition, 6-state BHMMs with 64 Bernoulli mixtures, grammar-scale-factor of 10 and word-insertion-penalty of 9), the recognition system achieved character recognition rate of 63.28% on the

test set. This achievement is promising in such challenging dataset of unconstrained handwritten text. As a comparison with the literature, the best character recognition accuracy rate reported in (Mahmoud et al. 2014) using the same dataset was 46.13%, using traditional statistical features and discrete HMMs.

Table 5.3: Character recognition rates using the sliding window and sliding window repositioning techniques

Window width with * indicates the sliding window repositioning is applied

Window Width	# Mixtures					
	2	**4**	**8**	**16**	**32**	**64**
3	33.38%	36.48%	40.44%	43.95%	47.04%	49.07%
5	34.15%	38.14%	42.25%	46.66%	49.69%	51.70%
7	35.13%	39.65%	44.58%	48.74%	51.28%	53.00%
9	35.33%	39.96%	45.03%	49.26%	51.23%	53.43%
9*	45.27%	49.66%	53.60%	56.67%	58.60%	59.44%

Table 5.4: The performance of Sliding Window Repositioning by tuning the grammar scale factor and word insertion penalty parameters of the Viterbi decoding

Grammar scale factor	Word insertion penalty	Character Accuracy %
1	1	59.44%
2	1	60.30%
3	1	66.99%
4	1	61.53%
5	1	62.01%
6	1	62.45%
7	1	62.74%
8	1	62.97%
9	1	63.16%
10	1	63.31%
10	6	63.37%
10	8	63.40%
10	**9**	**63.41%**
10	10	63.38%
11	1	63.31%
12	1	63.29%

5.3.3. Local Sampling

We have evaluated the local sampling approach on the four values of the sliding window width that we used in the previous section, viz., 3, 5, 7 and 9 pixels. The sliding window repositioning technique was applied to the 9-pixel width window. The results are shown in Table 5.5. The recognition rates of the local sampling approach are very close to that shown in Table 5.3 despite that the observations are of lower dimensionality. By adjusting the grammar-scale-factor and word-insertion-penalty parameters (Table 5.6), the 9-pixel width window with repositioning achieved 63.36% character accuracy on the validation set. Using the best configurations (32-pixel image height, 9-pixel window with reposition, 6-state BHMMs with 64 Bernoulli mixtures, grammar-scale-factor of 9 and word-insertion-penalty of 12), the local sampling approach achieved 63.34% character accuracy on the test set which is slightly better than that of the original (non-sampled) window.

Table 5.5: Character recognition rates of the local sampling approach

Window width with * indicates the repositioning is applied

Window Width	# Mixtures					
	2	**4**	**8**	**16**	**32**	**64**
3	33.77%	36.50%	40.20%	43.68%	46.56%	48.08%
5	33.98%	37.48%	41.63%	45.64%	48.85%	50.90%
7	34.35%	38.41%	43.47%	47.61%	50.75%	52.34%
9	34.66%	39.51%	44.19%	48.38%	51.37%	53.28%
9*	44.72%	49.28%	53.03%	56.15%	58.27%	59.03%

Table 5.6: The performance of local sampling by tuning the grammar scale factor and word insertion penalty parameters of the Viterbi decoding

Grammar scale factor	Word insertion penalty	Character Accuracy %
1	1	59.03%
2	1	60.14%
3	1	61.07%
4	1	61.72%
5	1	62.17%
6	1	62.58%
7	1	62.84%
8	1	62.99%
9	1	63.01%
9	4	63.17%
9	8	63.29%
9	10	63.32%
9	11	63.34%
9	**12**	**63.36%**
9	13	63.34%
10	1	63.01%
11	1	62.90%

5.3.4. Local Cell Layers

Similarly, the local cell layers approach was evaluated on the same four window sizes and the repositioning technique was applied to the 9-pixel width window. The character recognition accuracies are shown in Table 5.7. Comparing these results with the results in Table 5.3, we notice that the local cell layers improved the character recognition accuracy rates of small width windows (3- and 5-pixels windows). However, the wider windows achieved lower performance. This is attributed to the large reduction in the observation dimensionality of the wide windows. By adjusting the grammar-scale-factor and word-insertion-penalty parameters (Table 5.8), the 9-pixel width window with repositioning achieved 61.92% character recognition accuracy on the validation set using models of 64 mixtures per state. Using the best configurations (32-pixel image height, 9-pixel window with reposition, 6-state BHMMs with 64 Bernoulli mixtures, grammar-scale-factor of 6 and word-insertion-penalty of 6), the local sampling approach character achieved accuracy rate of 61.56% on the test set, which is an improvement given that the observation dimensionality is less than half.

Table 5.7: Character recognition rates using the local cell layers approach

Window width with * indicates the repositioning is applied

Window Width	# Mixtures					
	2	**4**	**8**	**16**	**32**	**64**
3	33.57%	36.56%	40.76%	44.42%	47.95%	49.84%
5	34.65%	38.62%	43.03%	47.02%	49.90%	52.02%
7	35.08%	39.55%	44.29%	48.14%	50.81%	52.51%
9	34.62%	38.67%	43.33%	47.34%	50.23%	51.91%
9*	42.91%	46.81%	50.88%	53.98%	56.29%	57.49%

Table 5.8: The performance of local cell layers by tuning the grammar scale factor and word insertion penalty parameters of the Viterbi decoding

Grammar scale factor	Word insertion penalty	Character Accuracy %
1	1	57.49%
2	1	59.15%
3	1	60.27%
4	1	61.02%
5	1	61.42%
6	1	61.44%
6	2	61.69%
6	4	61.90%
6	**6**	**61.92%**
6	7	61.88%
6	8	61.85%
7	1	61.27%
8	1	61.01%
9	1	60.61%

Besides the recognition accuracy, the two approaches are computationally fast due to lower dimensionality of the observation. In Table 5.9 we show the execution time of a single iteration of the Expectation-Maximization algorithm and in the evaluation on the validation set for the original system and the two proposed approaches. The low-dimensional observations dramatically speedup the training and evaluation implementations. For instance, the local cell layers approach gained 2.11X execution speedup than the original system in training and 1.79X in evaluation.

Table 5.9: Execution time (in hours) of a single iteration of training and in evaluation on the validation set by the three approaches

	Window Repositioning	**Local Sampling**	**Local Cell Layers**
Training	5.4425	4.0694	2.5778
Evaluation	1.3081	1.0492	0.7303

5.4. Conclusions

In this chapter, we have conducted extensive experimental evaluation on KHATT database using a BHMMs-based handwritten text recognition system in order to investigate the system performance on unconstrained Arabic handwritten text and to characterize the influence of various parameters in recognition accuracy. The results have shown that the system with the sliding window and window repositioning techniques can achieve prominent recognition accuracies with models of relatively few states. Increasing the number of Bernoulli mixtures is crucial to achieve higher recognition accuracy as the large number of mixtures can suitably model the dependency and correlation between the image bits. The sliding window and sliding window repositioning techniques significantly

improve the performance as they aid in observing the contextual information and reducing the image distortion. Best character recognition accuracy of 63.28% was achieved by using 32-pixel height images, 9-pixel width window with reposition and 6-state BHMMs with 64 Bernoulli mixtures.

To enhance the quality of the binary observations produced by the sliding window technique, we have proposed two approaches, coined as the local sampling and the local cell layers. Both approaches significantly reduce the observation dimensionality. In addition, the local cell layers approach imposes localization in the observations. The local sampling approach achieved 63.34% character recognition accuracy rate on KHATT database using the abovementioned configurations. This is slightly better than that achieved by the original system, although the observation dimensionality is two-thirds those of the original system.

The local cell layers achieved higher recognition performance with relatively small sliding windows due to its ability to impose spatial localization in the observations. Its recognition performance with wider windows, however, is less than that of the original system. This is attributed to the large reduction in the observation dimensionality of the wide windows. Best character recognition accuracy achieved on KHATT database is 61.56%. Despite the lower recognition performance, the local cell layers approach is computationally efficient in training and evaluation. It gained 2.11X execution speedup than the original approach in training and 1.79X in evaluation.

CHAPTER 6

CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

This chapter provides a summary of the contributions and findings of this dissertation and highlights the possible directions for extension. Section 6.1 provides conclusion remarks and highlights the main achievements of this dissertation. Section 6.2 provides directions for future research.

6.1. Conclusions

The main objective of this dissertation was to adapt the Bag-of-Features (BoF) framework to the recognition of open-vocabulary unconstrained Arabic handwritten text. The adaptation is based on exploiting writing context and the characteristics of Arabic handwritten text images. To pursue this objective, we established a baseline for BoF framework with options that have impact on the quality of the produced features. The baseline was evaluated on the recognition of isolated Arabic handwritten digits and subwords where it achieved state-of-the-art accuracies on two public datasets. The results provided evidence of the suitability of the framework to handwriting recognition.

We proceeded by enhancing the first stage of the framework baseline based on the characteristics of handwritten text. We have proposed two novel versions of SIFT that achieve the discriminative power of SIFT and are computationally efficient with half the size of the SIFT descriptors. The two versions achieved comparable recognition accuracies to the original SIFT. In addition to the recognition accuracies, the two versions take less

time to compute, and due to their lower dimensionality, the clustering and quantization phases became computationally efficient.

The approaches proposed to improve SIFT could be applied to other local descriptors that rely on gradient magnitude and orientation like the Histogram of Oriented Gradient. Many formats of gradient features suffer from the higher dimensionality of the produced feature vectors and resorted to different approaches for dimensionality reduction. Our approaches exploited the characteristics of handwritten text in order to produce discriminative low-dimensional histograms of gradient magnitude and orientation.

We utilized Gabor filter response features in implementing the first stage of the framework baseline, as these features previously achieved prominent accuracies in recognizing isolated handwritten digits. Although the recognition accuracies were lower, however, preparing low-level features to local descriptor layout significantly improve their discriminative power.

We expect the format of local descriptors would significantly improve other low-level features that rely on the visual appearance of handwritten text like the pixel intensities, wavelet transform and discrete cosine transform. Preparing these features into the format of local descriptors has two advantages. First, their discriminative power would be increased. Second, they could be easily integrated with BoF framework in order to produce mid-level features of better performance.

In order to apply the framework to the recognition of open-vocabulary unconstrained Arabic handwritten text, we integrated the framework baseline with a handwriting recognition system based on Hidden Markov Models (HMMs). Extensive experimental

evaluation was carried on KHATT database. The first stage of the framework baseline was adapted to the nature of the sliding window strategy. The adaptation enabled the system to achieve character accuracy of 48.40% on the validation set which is higher than the published results on the same set using the same recognizer trained using traditional statistical features. The writing baseline of the Arabic text was utilized to impose localization in the BoF representation. This addition achieved character recognition accuracy of 52.50% on the test set. The writing baseline also inspired us to utilize multi-stream HMMs in a novel approach that significantly improved the recognition accuracy. Including the abovementioned enhancements, the recognition system achieved character recognition accuracy of 64.30% on the test set which is promising as the dataset is challenging with unconstrained natural handwritten text.

For the sake of comparison, we implemented a segmentation-free handwriting recognition system based on Bernoulli Hidden Markov Models (BHMMs). Our implementation achieved character recognition accuracy of 63.28% on KHATT database which is comparable to the results we achieved using the BoF representations with the traditional HMMs. The two approaches we proposed to reduce the dimensionality of the binary observations and to impose spatial localization achieved comparable recognition accuracies, in addition to the computational efficiency they showed.

This work indicates the power of BoF framework in producing robust representations for handwriting recognition. It indicates also that exploiting the context and characteristics Arabic handwritten text images as well as the carful adaptation of the framework to the recognition system are crucial to achieve improved recognition accuracies.

6.2. Future Directions

The work presented in this dissertation can be extended in different directions. Some extensions are highlighted in the following points:

1. **Enhancing the performance of the two proposed versions of SIFT:** in this work, we eliminated the Gaussian pre-smoothing step and utilized the basic derivative filters for computing the gradient magnitude and orientation. Although the basic derivative filters achieved satisfactory performance, using advanced filters like Prewitt or Sobel would improve the performance as these two filters apply local smoothing in the orthogonal direction to the derivative direction. Further, lookup tables still can be applied for computing the gradient magnitude and orientation, although they will become larger, since these filters utilize the 8-neighbors instead of the left/right or top/bottom neighbors utilized in the basic filters.
2. **Enhancing the performance of the Gabor descriptors:** our experimentation results showed that normalizing the final vector to unit length has great impact in the recognition accuracies. Recently, Gabor filter response features were utilized in developing SIFT-like descriptor coined the Biologically Inspired Local Descriptor (BILD) (Zhang et al. 2014). In this work, several normalizations inspired by "*the visual information processing mechanism of ventral pathway in human brain*" were proposed. Applying these normalizations to the Gabor descriptors we developed in this work would improve their performance.
3. **Exploring more discriminative low-level features:** the performance of the two-stage feature learning frameworks could be enhanced by using strong low-level features (Coates & Ng 2011b). There are many low-level features proposed in the

format of local descriptors that claimed superior performance than SIFT in visual object recognition and matching (Mikolajczyk & Schmid 2005) (Bay et al. 2008) (Heikkilä et al. 2009) (Zhang et al. 2014). Exploring these algorithms for handwriting recognition and utilizing those that show prominent performance to BoF framework would enhance the quality of the produced features. Further, the characteristics of the text images and handwritten text could be also utilized in improving these algorithms as we did with SIFT in this work.

4. **Utilizing deep architectures:** in this work, we produced mid-level features augmented by context and the characteristics of Arabic handwritten text. These augmented features could be utilized in deep architectures for producing higher level discriminative features. Several frameworks are suitable for building higher features upon them. The *hyperfeatures* framework (Agarwal & Triggs 2006) and the multi-stage framework presented in (Coates & Ng 2011a) could be utilized to produce higher level features in unsupervised manner. Supervised frameworks similar to the one presented in (Wang et al. 2012) could be also utilized to produce higher level features in supervised manner for isolated digits and words when labeled training samples are available.
5. **Applying dictionaries and word-level language models:** Word level and dictionaries can be utilized to improve the recognition accuracies of the recognition systems. The utilization of these techniques enabled HMM-based systems to achieved higher recognition rates on KHATT database (Hamdani et al. 2013) (Stahlberg & Vogel 2015).

REFERENCES

AbdulKader, A., 2008. A Two-Tier Arabic Offline Handwriting Recognition Based on Conditional Joining Rules. In D. S. Doermann & S. Jaeger, eds. *Arabic and Chinese Handwriting Recognition*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 70–81.

Abuhaiba, I.S.I., Mahmoud, S.A. & Green, R.J., 1994. Recognition of handwritten cursive Arabic characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6), pp.664–672.

Agarwal, A. & Triggs, B., 2006. Hyperfeatures – Multilevel Local Coding for Visual Recognition. In A. Leonardis, H. Bischof, & A. Pinz, eds. *European Conference on Computer Vision–ECCV 2006*. Springer-Verlag Berlin Heidelberg, pp. 30–43.

Ahmad, I., Fink, G.A. & Mahmoud, S.A., 2014. Improvements in Sub-character HMM Model Based Arabic Text Recognition. In *14th International Conference on Frontiers in Handwriting Recognition*. Crete, Greece: IEEE, pp. 537–542.

Ahmad, I., Mahmoud, S.A. & Fink, G.A., 2016. Open-Vocabulary Recognition of Machine-Printed Arabic Text Using Hidden Markov Models. *Pattern Recognition*, 51, pp.97–111.

Ahmad, I., Mahmoud, S.A. & Parvez, M.T., 2012. Printed Arabic Text Recognition. In V. Märgner & H. El-Abed, eds. *Guide to OCR for Arabic Scripts*. London: Springer London, pp. 147–168.

Al-dmour, A. & Abuhelaleh, M., 2016. Arabic Handwritten Word Category Classification Using Bag of Features. *Journal of Theoretical and Applied Information Technology*, 89(2), pp.320–328.

Al-Jamimi, H.A. & Mahmoud, S.A., 2010. Arabic Character Recognition Using Gabor Filters. In T. Sobh, ed. *Innovations and Advances in Computer Sciences and Engineering*. Dordrecht, Netherlands: Springer Science+Business Media B.V., pp. 113–118.

Al-Muhtaseb, H.A., Mahmoud, S.A. & Qahwaji, R.S., 2008. Recognition of Off-line Printed Arabic Text using Hidden Markov Models. *Signal Processing*, 88(12), pp.2902–2912.

Al-Ohali, Y., Cheriet, M. & Suen, C.Y., 2004. Databases for Recognition of Handwritten Arabic Cheques. *Pattern Recognition*, 36, pp.111–121.

Aldavert, D. et al., 2015. A Study of Bag-of-Visual-Words Representations for Handwritten Keyword Spotting. *International Journal on Document Analysis and Recognition (IJDAR)*, 18(3), pp.223–234.

Aldavert, D. et al., 2013. Integrating Visual and Textual Cues for Query-by-String Word Spotting. In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA, pp. 511–515.

Alkhateeb, J.H. et al., 2011. Offline Handwritten Arabic Cursive Text Recognition using Hidden Markov Models and Re-ranking. *Pattern Recognition Letters*, 32(8), pp.1081–1088.

AlKhateeb, J.H. et al., 2008. Word-based Handwritten Arabic Scripts Recognition using DCT Features and Neural Network Classifier. In *5th International Multi-Conference on Systems, Signals and Devices*. Amman, Jordan: IEEE, pp. 1–5.

Alkhoury, I., Giménez, A. & Juan, A., 2012. Arabic Handwriting Recognition Using Bernoulli HMMs. In V. Märgner & H. El-Abed, eds. *Guide to OCR for Arabic Scripts*. London: Springer London, pp. 255–272.

Alma'adeed, S., Higgins, C. & Elliman, D., 2004. Off-line Recognition of Handwritten Arabic Words using Multiple Hidden Markov Models. *Knowledge-Based Systems*, 17(2–4), pp.75–79.

Anil, R. et al., 2015. Convolutional Neural Networks for the Recognition of Malayalam Characters. In S. C. Satapathy, S. K. Udgata, & B. N. Biswal, eds. *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA 2014)*. Bhubaneswar, India: Springer International Publishing, pp. 493–500.

Arandjelovic, R. & Zisserman, A., 2013. All About VLAD. In *IEEE Conference on Computer Vision and Pattern Recognition*. Portland, Oregon, USA: IEEE, pp. 1578–1585.

Augustin, E. et al., 2006. RIMES Evaluation Campaign for Handwritten Mail Processing. In *Workshop on Frontiers in Handwriting Recognition*. La Baule, France, pp. 1–5.

Avila, S. et al., 2013. Pooling in Image Representation: The Visual Codeword Point of View. *Computer Vision and Image Understanding*, 117(5), pp.453–465.

Awaida, S. & Mahmoud, S.A., 2014. Automatic Check Digits Recognition for Arabic Using Multi-Scale Features, HMM and SVM Classifiers. *British Journal of Mathematics & Computer Science*, 4(17), pp.2521–2535.

Bailly, A. et al., 2015. Bag-of-Temporal-SIFT-Words for Time Series Classification. In *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data*. Porto, Portugal, pp. 1–7.

Bay, H. et al., 2008. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3), pp.346–359.

Bayat, A.B., 2014. Sparse Representation-based Classification of Farsi Handwritten Digits Using Fisher Discrimination Criterion and Local Linear Embedding (LLE). *Journal of Pattern Recognition and Intelligent Systems*, 2(4), pp.56–65.

Beaudet, P.R., 1978. Rotationally Invariant Image Operators. In *4th International Conference on Pattern Recognition*. Kyoto, Japan, pp. 579–583.

Belongie, S., Malik, J. & Puzicha, J., 2002. Shape Matching and Object Recognition using Shape Contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4), pp.509–522.

Bengio, Y., 2009. Learning Deep Architectures for AI. *Foundations and Trends® in Machine Learning*, 2(1), pp.1–127.

Bengio, Y., Courville, A. & Vincent, P., 2013. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), pp.1798–1828.

Bhattacharya, U. & Chaudhuri, B.B., 2003. A Majority Voting Scheme for Multiresolution Recognition of Handprinted Humerals. In *Seventh International Conference on Document Analysis and Recognition (ICDAR'03)*. Washington, DC, USA: IEEE Comput. Soc, pp. 16–20.

Bluche, T., Louradour, J., et al., 2014. The A2iA Arabic Handwritten Text Recognition System at the Open HaRT2013 Evaluation. In *11th IAPR International Workshop on Document Analysis Systems*. Tours, France: IEEE, pp. 161–165.

Bluche, T., Ney, H. & Kermorvant, C., 2014. A Comparison of Sequence-Trained Deep Neural Networks and Recurrent Neural Networks Optical Modeling for Handwriting Recognition. In L. Besacier, A.-H. Dediu, & C. Martín-Vide, eds. *Statistical Language and Speech Processing*. Springer International Publishing, pp. 199–210.

Bluche, T., Ney, H. & Kermorvant, C., 2013a. Feature Extraction with Convolutional Neural Networks for Handwritten Word Recognition. In *12th International Conference on Document Analysis and Recognition*. Washington, DC, USA: IEEE, pp. 285–289.

Bluche, T., Ney, H. & Kermorvant, C., 2013b. Tandem HMM with Convolutional Neural Network for Handwritten Word Recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. Vancouver, BC, Canada: IEEE, pp. 2390–2394.

Bosch, A., Zisserman, A. & Munoz, X., 2007. Image Classification using Random Forests and Ferns. In *IEEE 11th International Conference on Computer Vision*. Rio de Janeiro, Brazil: IEEE, pp. 1–8.

Boureau, Y.-L., Bach, F., et al., 2010. Learning Mid-Level Features for Recognition. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA USA: IEEE, pp. 2559–2566.

Boureau, Y.-L., Ponce, J. & LeCun, Y., 2010. A Theoretical Analysis of Feature Pooling in Visual Recognition. In *27th International Conference on Machine Learning (ICML-10)*. Haifa, pp. 111–118.

Breuel, T.M. et al., 2013. High-Performance OCR for Printed English and Fraktur Using LSTM Networks. In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA: IEEE, pp. 683–687.

Brown, M., Hua, G. & Winder, S., 2011. Discriminative Learning of Local Image Descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1), pp.43–57.

Brunelli, R. & Poggio, T., 1993. Face Recognition: Features Versus Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10), pp.1042–1052.

Busch, A., Boles, W.W. & Sridharan, S., 2005. Texture for Script Identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11), pp.1720–1732.

Calonder, M. et al., 2012. BRIEF: Computing a Local Binary Descriptor Very Fast. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7), pp.1281–1298.

Chatfield, K. et al., 2011. The Devil Is In The Details: An Evaluation of Recent Feature Encoding Methods. In *The 22nd British Machine Vision Conference*. Dundee, UK, pp. 1–12.

Chellapilla, K., Puri, S. & Simard, P., 2006. High Performance Convolutional Neural Networks for Document Processing. In *Tenth International Workshop on Frontiers in Handwriting Recognition*. La Baule, France, pp. 1–6.

Chellapilla, K., Shilman, M. & Simard, P., 2006. Optimally Combining A Cascade of Classifiers. In K. Taghva & X. Lin, eds. *Document Recognition and Retrieval XIII, 60670Q*. San Jose, CA: International Society for Optics and Photonics, pp. 1–8.

Chellapilla, K. & Simard, P., 2006. A New Radical Based Approach to Offline Handwritten East-Asian Character Recognition. In *Tenth International Workshop on Frontiers in Handwriting Recognition*. La Baule, France, pp. 1–6.

Chen, J. et al., 2010. Gabor Features for Offline Arabic Handwriting Recognition. In *Proceedings of the 8th IAPR International Workshop on Document Analysis Systems (DAS '10)*. DAS '10. Cambridge, MA, USA: ACM Press, pp. 53–58.

Cheriet, M. et al., 2007. Arabic Cheque Processing System: Issues and Future Trends. In B. B. Chaudhuri, ed. *Digital Document Processing*. London: Springer London, pp. 213–234.

Chherawala, Y., Roy, P.P. & Cheriet, M., 2013. Feature Design for Offline Arabic Handwriting Recognition: Handcrafted vs Automated? In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA, pp. 290–294.

Christlein, V., Bernecker, D. & Angelopoulou, E., 2015. Writer Identification using VLAD Encoded Contour-Zernike Moments. In *13th International Conference on Document Analysis and Recognition (ICDAR 2015)*. Tunis, Tunisia: IEEE, pp. 906–910.

Cireşan, D.C. et al., 2011. Convolutional Neural Network Committees for Handwritten Character Classification. In *11th International Conference on Document Analysis and Recognition (ICDAR 2011)*. Beijing, China: IEEE, pp. 1135–1139.

Cireşan, D.C. et al., 2010. Deep, Big, Simple Neural Nets for Handwritten Digit Recognition. *Neural Computation*, 22(12), pp.3207–3220.

Cireşan, D.C., Meier, U. & Schmidhuber, J., 2012. Multi-Column Deep Neural Networks for Image Classification. In *IEEE Conference on Computer Vision and Pattern Recognition*. Providence, RI, USA: IEEE, pp. 3642–3649.

Cireşan, D.C., Meier, U. & Schmidhuber, J., 2012. Transfer Learning for Latin and Chinese characters with Deep Neural Networks. In *2012 International Joint Conference on Neural Networks (IJCNN)*. Brisbane, Australia: IEEE, pp. 1–6.

Cireşan, D.C. & Schmidhuber, J., 2013. Multi-Column Deep Neural Networks for Offline Handwritten Chinese Character Classification. *arXiv preprint arXiv:1309.0261*, pp.1–5.

Coates, A., 2012. *Demystifying Unsupervised Feature Learning*. Ph.D. Thesis, Stanford University.

Coates, A., Carpenter, B., et al., 2011. Text Detection and Character Recognition in Scene Images with Unsupervised Feature Learning. In *11th International Conference on Document Analysis and Recognition (ICDAR 2011)*. Beijing, China: Ieee, pp. 440–445.

Coates, A., Lee, H. & Ng, A.Y., 2011. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In *International Conference on AI and Statistics*. Ft. Lauderdale, FL, USA, pp. 215–223.

Coates, A. & Ng, A.Y., 2012. Learning Feature Representations with K-Means. In G. Montavon, G. Orr, & K.-R. Müller, eds. *Neural Networks: Tricks of the Trade*. Springer Berlin Heidelberg, pp. 561–580.

Coates, A. & Ng, A.Y., 2011a. Selecting Receptive Fields in Deep Networks. In J. Shawe-Taylor et al., eds. *Advances in Neural Information Processing Systems 24 (NIPS 2011)*. Granada Spain: Curran Associates, Inc., pp. 2528–2536.

Coates, A. & Ng, A.Y., 2011b. The Importance of Encoding Versus Training with Sparse Coding and Vector Quantization. In Lise Getoor & Tobias Scheffer, eds. *28th International Conference on Machine Learning (ICML-11)*. ICML '11. Washington, USA: ACM, pp. 921–928.

Csurka, G. et al., 2004. Visual Categorization with Bags of Keypoints. In *Workshop on statistical learning in computer vision (ECCV)*. Prague, Czech Republic, pp. 1–22.

Dehghan, M. et al., 2001. Handwritten Farsi (Arabic) Word Recognition: A Holistic Approach using Discrete HMM. *Pattern Recognition*, 34(5), pp.1057–1065.

Del-Agua, M.A. et al., 2014. The transLectures-UPV Toolkit. In *Advances in Speech and Language Technologies for Iberian Languages*. Gran Canaria, Spain, pp. 269–278.

Deng, L., 2014. A Tutorial Survey Of Architectures, Algorithms, And Applications For Deep Learning. *APSIPA Transactions on Signal and Information Processing*, 3, pp.1–29.

Deng, L. et al., 2010. Binary Coding of Speech Spectrograms Using a Deep Auto-encoder. In *Interspeech*. Makuhari, Chiba, Japan, pp. 1692–1695.

Deng, L. & Yu, D., 2014. Deep Learning: Methods and Applications. *Foundations and Trends® in Signal Processing*, 7(3–4), pp.197–387.

Doetsch, P. et al., 2012. Comparison of Bernoulli and Gaussian HMMs Using a Vertical Repositioning Technique for Off-Line Handwriting Recognition. In *2012 International Conference on Frontiers in Handwriting Recognition*. Bari, Italy: IEEE, pp. 3–7.

Dreuw, P. et al., 2009. Writer Adaptive Training and Writing Variant Model Refinement for Offline Arabic Handwriting Recognition. In *10th International Conference on Document Analysis and Recognition (ICDAR 2009)*. Catalonia, Spain: IEEE, pp. 21–25.

Dreuw, P., Jonas, S. & Ney, H., 2008. White-Space Models for Offline Arabic Handwriting Recognition. In *2008 19th International Conference on Pattern Recognition*. Tampa, FL, USA: IEEE, pp. 1–4.

Duda, R.O., Hart, P.E. & Stork, D.G., 2001. *Pattern Classification* Second Edi., John Wiley and Sons, Inc.

Echi, A.K. & Saidani, A., 2014. How to Separate Between Machine-Printed / Handwritten and Arabic / Latin Words ? *Electronic Letters on Computer Vision and Image Analysis*, 13(1), pp.1–16.

Elleuch, M., Maalej, R. & Kherallah, M., 2016. A New Design Based-SVM of the CNN Classifier Architecture with Dropout for Offline Arabic Handwritten Recognition. *Procedia Computer Science*, 80, pp.1712–1723.

Elleuch, M., Tagougui, N. & Kherallah, M., 2015a. Deep Learning for Feature Extraction of Arabic Handwritten Script. In G. Azzopardi & N. Petkov, eds. *Computer Analysis of Images and Patterns*. Springer International Publishing, pp. 371–382.

Elleuch, M., Tagougui, N. & Kherallah, M., 2015b. Towards Unsupervised Learning for Arabic Handwritten Recognition Using Deep Architectures. In S. Arik et al., eds. *Neural Information Processing*. Springer International Publishing, pp. 363–372.

Elms, A.J. & Illingworth, J., 1995. Modelling Polyfont Printed Characters with HMMs and a Shift Invariant Hamming Distance. In *Proceedings of 3rd International Conference on Document Analysis and Recognition (ICDA'95)*. Montreal, Que., Canada: IEEE Comput. Soc. Press, pp. 504–507.

Elzobi, M. et al., 2013. A Hidden Markov Model-Based Approach with an Adaptive Threshold Model for Off-Line Arabic Handwriting Recognition. In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA: IEEE, pp. 945–949.

Elzobi, M. et al., 2012. Arabic Handwriting Recognition using Gabor Wavelet Transform and SVM. In *2012 IEEE 11th International Conference on Signal Processing*. Beijing, China: IEEE, pp. 2154–2158.

Elzobi, M. et al., 2014. Gabor Wavelet Recognition Approach for Off-Line Handwritten Arabic Using Explicit Segmentation. In R. S. Choras, ed. *Image Processing and Communications Challenges 5*. Springer International Publishing, pp. 245–254.

En, S. et al., 2016. A Scalable Pattern Spotting System for Historical Documents. *Pattern Recognition*, 54, pp.149–161.

Erhan, D. et al., 2010. Why Does Unsupervised Pre-training Help Deep Learning? *The Journal of Machine Learning Research*, 11, pp.625–660.

Fiel, S. & Sablatnig, R., 2013. Writer Identification and Writer Retrieval Using the Fisher Vector on Visual Vocabularies. In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA, pp. 545–549.

Fiel, S. & Sablatnig, R., 2012. Writer Retrieval and Writer Identification Using Local Features. In *2012 10th IAPR International Workshop on Document Analysis Systems*. Queensland, Australia: IEEE, pp. 145–149.

Fink, G.A., 2014. *Markov Models for Pattern Recognition* Second Edi., Berlin, Heidelberg: Springer Berlin Heidelberg.

Fink, G.A. & Plotz, T., 2005. On Appearance-Based Feature Extraction methods for Writer-Independent Handwritten Text Recognition. In *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*. Washington, DC, USA: IEEE, p. 1070–1074 Vol. 2.

Fink, G.A., Rothacker, L. & Grzeszick, R., 2014. Grouping Historical Postcards Using Query-by-Example Word Spotting. In *14th International Conference on Frontiers in Handwriting Recognition*. Crete, Greece: IEEE, pp. 470–475.

Frinken, V. & Bunke, H., 2010. Self-training for Handwritten Text Line Recognition. In *Proceedings of the 15th Iberoamerican congress conference on Progress in pattern recognition, image analysis, computer vision, and applications*. São Paulo, Brazil: Springer-Verlag Berlin, Heidelberg, pp. 104–112.

Fukushima, K., 1980. Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. *Biological Cybernetics*, 36(4), pp.193–202.

Gandhi, A. & Jawahar, C.V., 2013. Detection of Cut-and-Paste in Document Images. In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA, pp. 653–657.

van Gemert, J.C. et al., 2010. Visual Word Ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7), pp.1271–1283.

Giménez, A. et al., 2011. Discriminative Bernoulli Mixture Models for Handwritten Digit Recognition. In *11th International Conference on Document Analysis and Recognition (ICDAR 2011)*. Beijing, China: IEEE, pp. 558–562.

Giménez, A. & Juan, A., 2009a. Bernoulli HMMs at Subword Level for Handwritten Word Recognition. In H. J. Araújo, A. M. Mendonça, & A. J. Pinho, eds. *Pattern Recognition and Image Analysis*. Springer Berlin Heidelberg, pp. 497–504.

Giménez, A. & Juan, A., 2009b. Embedded Bernoulli Mixture HMMs for Continuous Handwritten Text Recognition. In X. Jiang & N. Petkov, eds. *Computer Analysis of*

Images and Patterns. Springer Berlin Heidelberg, pp. 197–204.

Giménez, A. & Juan, A., 2009c. Embedded Bernoulli Mixture HMMs for Handwritten Word Recognition. In *10th International Conference on Document Analysis and Recognition (ICDAR 2009)*. Catalonia, Spain: IEEE, pp. 896–900.

Giménez, A., Khoury, I. & Juan, A., 2010. Windowed Bernoulli Mixture HMMs for Arabic Handwritten Word Recognition. In *2010 12th International Conference on Frontiers in Handwriting Recognition*. IEEE, pp. 533–538.

Goodfellow, I., Bengio, Y. & Courville, A., 2016. *Deep Learning*, Book in preparation for MIT Press (http://www.deeplearningbook.org).

Grauman, K. & Leibe, B., 2011. *Visual Object Recognition*, San Francisco: Morgan & Claypool.

Graves, A. et al., 2009. A Novel Connectionist System for Improved Unconstrained Handwriting Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5), pp.855–868.

Graves, A., 2012. Offline Arabic Handwriting Recognition with Multidimensional Recurrent Neural Networks. In V. Märgner & H. El-Abed, eds. *Guide to OCR for Arabic Scripts*. London: Springer London, pp. 297–313.

Graves, A. & Schmidhuber, J., 2009. Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks. In Y. Bengio et al., eds. *Advances in Neural Information Processing Systems 22 (NIPS 2009)*. Vancouver, B.C., Canada, pp. 545–552.

Grosse, R. et al., 2012. Shift-Invariance Sparse Coding for Audio Classification. *arXiv preprint arXiv:1206.5241*, pp.1–10.

Gupta, R., Patil, H. & Mittal, A., 2010. Robust Order-Based Methods for Feature Description. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. New York, USA: IEEE, pp. 334–341.

Haboubi, S. et al., 2009. Invariant Primitives for Handwritten Arabic Script: A Contrastive Study of Four Feature Sets. In *10th International Conference on Document Analysis and Recognition (ICDAr 22009)*. Catalonia, Spain, pp. 691–697.

Hamdani, M., Mousa, A.E.-D. & Ney, H., 2013. Open Vocabulary Arabic Handwriting Recognition Using Morphological Decomposition. In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA: IEEE, pp. 280–284.

Hammerla, N.Y. et al., 2010. Towards Feature Learning for HMM-based Offline Handwriting Recognition. In *First International Workshop on Frontiers of Arabic Handwriting Recognition*. Istanbul, Turkey, pp. 27–32.

Harris, C. & Stephens, M., 1988. A Combined Corner and Edge Detector. In *Alvey Vision Conference*. Manchester, UK: Alvey Vision Club, pp. 147–151.

Heikkilä, M., Pietikäinen, M. & Schmid, C., 2009. Description of Interest Regions with Local Binary Patterns. *Pattern Recognition*, 42(3), pp.425–436.

Helli, B. & Moghaddam, M.E., 2010. A Text-Independent Persian Writer Identification Based on Feature Relation Graph (FRG). *Pattern Recognition*, 43(6), pp.2199–2209.

Hinton, G., Osindero, S. & Teh, Y.-W., 2006. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18(7), pp.1527–1554.

Hinton, G.E. & Zemel, R.S., 1994. Autoencoders, Minimum Description Length, and Helmholtz Free Energy. In J. D. Cowan, G. Tesauro, & J. Alspector, eds. *Advances in Neural Information Processing Systems 6 (NIPS 1994)*. Morgan-Kaufmann, pp. 3–10.

Hu, J., Peng, X. & Fu, C., 2015. A Comparison of Feature Description Algorithms. *Optik - International Journal for Light and Electron Optics*, 126(2), pp.274–278.

Huang, M. et al., 2015. Local Image Region Description using Orthogonal Symmetric Local Ternary Pattern. *Pattern Recognition Letters*, 54, pp.56–62.

Hubel, D.H. & Wiesel, T.N., 1962. Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex. *The Journal of physiology*, 160(1), pp.106–154.

Hyvärinen, A. & Oja, E., 2000. Independent Component Analysis: Algorithms and Applications. *Neural Networks*, 13(4–5), pp.411–430.

Ionescu, R.T. & Popescu, M., 2015. PQ Kernel: A Rank Correlation Kernel for Visual Word Histograms. *Pattern Recognition Letters*, 55, pp.51–57.

Ionescu, R.T., Popescu, M. & Grozea, C., 2013. Local Learning to Improve Bag of Visual Words Model for Facial Expression Recognition. In *Workshop on challenges in representation learning, ICML*. Georgia, USA, pp. 1–6.

Jain, R. & Doermann, D., 2011. Offline Writer Identification Using K-Adjacent Segments. In *2011 11th International Conference on Document Analysis and Recognition*. Beijing, China: IEEE, pp. 769–773.

Jarrett, K. et al., 2009. What Is the Best Multi-Stage Architecture for Object Recognition? In *2009 IEEE 12th International Conference on Computer Vision*. Kyoto, Japan: IEEE, pp. 2146–2153.

Jayech, K., Mahjoub, M.A. & Ben Amara, N.E., 2016. Synchronous Multi-Stream Hidden Markov Model for Offline Arabic Handwriting Recognition Without Explicit Segmentation. *Neurocomputing*, 214, pp.958–971.

Jianxin, W. & Rehg, J.M., 2009. Beyond the Euclidean Distance: Creating Effective Visual codebooks using the Histogram Intersection Kernel. In *2009 IEEE 12th International Conference on Computer Vision*. Kyoto, Japan: IEEE, pp. 630–637.

Jolliffe, I., 2002. *Principal Component Analysis*, Wiley Online Library.

Juan, A. & Vidal, E., 2004. Bernoulli Mixture Models for Binary Images. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)*. Washington, DC, USA: IEEE, pp. 367–370.

Jurie, F. & Triggs, B., 2005. Creating Efficient Codebooks for Visual Recognition. In *Tenth IEEE International Conference on Computer Vision (ICCV'05)*. Beijing, China: IEEE, p. 604–610 Vol. 1.

Ke, Y. & Sukthankar, R., 2004. PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004)*. Washington, DC, USA: IEEE, pp. 506–513.

Kessentini, Y., Paquet, T. & Ben Hamadou, A., 2010. Off-line Handwritten Word Recognition using Multi-stream Hidden Markov Models. *Pattern Recognition Letters*, 31(1), pp.60–70.

Khorsheed, M.S., 2007. Offline Recognition of Omnifont Arabic Text using the HMM ToolKit (HTK). *Pattern Recognition Letters*, 28(12), pp.1563–1571.

Kim, I.-J. & Xie, X., 2015. Handwritten Hangul Recognition using Deep Convolutional Neural Networks. *International Journal on Document Analysis and Recognition (IJDAR)*, 18(1), pp.1–13.

Kingma, D.P. et al., 2014. Semi-Supervised Learning with Deep Generative Models. *arXiv preprint arXiv:1406.5298*, pp.1–9.

Koniusz, P. & Mikolajczyk, K., 2011. Spatial Coordinate Coding to Reduce Histogram Representations, Dominant Angle and Colour Pyramid Match. In *2011 18th IEEE International Conference on Image Processing*. Brussels, Belgium: IEEE, pp. 661–664.

Kozielski, M., Doetsch, P. & Ney, H., 2013. Improvements in RWTH's System for Off-Line Handwriting Recognition. In *12th International Conference on Document Analysis and Recognition*. Washington, DC, USA: IEEE, pp. 935–939.

Krizhevsky, A., Sutskever, I. & Hinton, G., 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances In Neural Information Processing Systems*. Stateline, NV, USA, pp. 1097–1105.

Law, M.T., Thome, N. & Cord, M., 2014. Bag-of-Words Image Representation: Key Ideas and Further Insight. In B. Ionescu et al., eds. *Fusion in Computer Vision, Advances in Computer Vision and Pattern Recognition*. Switzerland: Springer International Publishing, pp. 29–52.

Lazebnik, S., Schmid, C. & Ponce, J., 2006. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *Computer Vision and Pattern Recognition (CVPR'06)*. New York, NY, USA: IEEE, pp. 2169–2178.

Le, Q. V., 2013. Building High-level Features using Large Scale Unsupervised Learning. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. Vancouver, BC, Canada: IEEE, pp. 8595–8598.

LeCun, Y. et al., 1989. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4), pp.541–551.

LeCun, Y. et al., 1998. Gradient-based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11), pp.2278–2324.

LeCun, Y. et al., 1990. Handwritten Digit Recognition with a Back-propagation Network. In R. P. Lippmann, J. E. Moody, & D. S. Touretzky, eds. *Advances in Neural Information Processing Systems 3 (NIPS 1990)*. Morgan Kaufmann Publishers Inc., pp. 396–404.

LeCun, Y., 2012. Learning Invariant Feature Hierarchies. In A. Fusiello, V. Murino, & R. Cucchiara, eds. *Computer Vision – ECCV 2012*. Florence, Italy: Springer Berlin Heidelberg, pp. 496–505.

LeCun, Y., Bengio, Y. & Hinton, G., 2015. Deep learning. *Nature*, 521(7553), pp.436–444.

LeCun, Y., Bottou, L. & Bengio, Y., 1997. Reading Checks with Graph Transformer Networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-97)*. Munich, Germany, pp. 151–154.

Lindeberg, T., 1993. Detecting Salient Blob-like Image Structures and their Scales with a Scale-space Primal Sketch: A Method for Focus-of-Attention. *International Journal of Computer Vision*, 11(3), pp.283–318.

Liu, C. et al., 2013. Handwritten Character Recognition with Sequential Convolutional Neural Network. In *2013 International Conference on Machine Learning and Cybernetics*. Tianjin, China: IEEE, pp. 291–296.

Liu, L., Wang, L. & Liu, X., 2011. In defense of soft-assignment coding. In *2011 International Conference on Computer Vision (ICCV)*. Barcelona, Spain: IEEE, pp. 2486–2493.

Liwicki, M., Graves, A. & Bunke, H., 2012. Neural Networks for Handwriting Recognition. In M. R. Ogiela & L. C. Jain, eds. *Computational Intelligence Paradigms in Advanced Pattern Classification*. Springer Berlin Heidelberg, pp. 5–24.

López-Monroy, A.P. et al., 2016. Improving the BoVW via Discriminative Visual N-Grams and MKL Strategies. *Neurocomputing*, 175(Part A), pp.768–781.

Lowe, D.G., 2004. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2), pp.91–110.

Maalej, R. & Kherallah, M., 2016. Improving MDLSTM for Offline Arabic Handwriting Recognition Using Dropout at Different Positions. In A. E. P. Villa, P. Masulli, & A. J. P. Rivero, eds. *Artificial Neural Networks and Machine Learning – ICANN 2016*. Springer International Publishing, pp. 431–438.

Maalej, R., Tagougui, N. & Kherallah, M., 2016. Recognition of Handwritten Arabic Words with Dropout Applied in MDLSTM. In A. Campilho & F. Karray, eds. *Image Analysis and Recognition*. Springer International Publishing, pp. 746–752.

Mahmoud, S.A., 2008. Arabic (Indian) Handwritten Digits Recognition using Gabor-based Features. In *2008 International Conference on Innovations in Information Technology*. Kuwait: IEEE, pp. 683–687.

Mahmoud, S.A. et al., 2014. KHATT: An Open Arabic Offline Handwritten Text Database. *Pattern Recognition*, 47(3), pp.1096–1112.

Mahmoud, S.A., 2009. Recognition of Arabic (Indian) Check Digits using Spatial Gabor Filters. In *5th IEEE-GCC Conference & Exhibition*. Kuwait: IEEE, pp. 1–5.

Mahmoud, S.A. & Al-Khatib, W.G., 2010. Recognition of Arabic (Indian) Bank Check Digits using Log-Gabor Filters. *Applied Intelligence*, 35(3), pp.445–456.

Manjunath, B.S. & Ma, W.Y., 1996. Texture Features for Browsing and Retrieval of Large Image Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8), pp.837–842.

Margner, V. & El-Abed, H., 2010. ICFHR 2010 - Arabic Handwriting Recognition Competition. In *2010 12th International Conference on Frontiers in Handwriting Recognition*. Kolkata, India: IEEE, pp. 709–714.

Märgner, V., EL-Abed, H. & Pechwitz, M., 2006. Offline Handwritten Arabic Word Recognition Using HMM - a Character Based Approach without Explicit Segmentation. In *SDN06 Laurence Likforman-Sulem <hal-00112048>*. France: SDN06, pp. 259–264.

Marti, U.-V. & Bunke, H., 2002. The IAM-database: an English Sentence Database for Offline Handwriting Recognition. *International Journal on Document Analysis and Recognition*, 5(1), pp.39–46.

Mikolajczyk, K. et al., 2005. A Comparison of Affine Region Detectors. *International Journal of Computer Vision*, 65(1–2), pp.43–72.

Mikolajczyk, K. & Schmid, C., 2001. Indexing Based on Scale Invariant Interest Points. In *Proceedings Eighth IEEE International Conference on Computer Vision (ICCV 2001)*. Vancouver, BC, Canada: IEEE Comput. Soc, pp. 525–531.

Mikolajczyk, K. & Schmid, C., 2005. Performance Evaluation of Local Descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 27(10), pp.1615–30.

Mikolajczyk, K. & Schmid, C., 2004. Scale & Affine Invariant Interest Point Detectors. *International Journal of Computer Vision*, 60(1), pp.63–86.

Moravec, H.P., 1980. *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover, Tech. Report, Robotics Institute, Carnegie-Mellon University, pp.1-175*,

Movellan, J.R., 2008. Tutorial on Gabor Filters. , pp.1–23.

Nowak, E., Jurie, F. & Triggs, B., 2006. Sampling Strategies for Bag-of-Features Image Classification. In A. Leonardis, H. Bischof, & A. Pinz, eds. *Computer Vision – ECCV 2006*. Springer, Berlin, Heidelberg, pp. 490–503.

O'Hara, S. & Draper, B.A., 2011. Introduction to the Bag of Features Paradigm for Image Classification and Retrieval. *arXiv preprint arXiv:1101.3354*, pp.1–25.

Ojala, T., Pietikäinen, M. & Harwood, D., 1996. A Comparative Study of Texture Measures with Classification Based on Featured Distributions. *Pattern Recognition*, 29(1), pp.51–59.

Ojala, T., Pietikainen, M. & Maenpaa, T., 2002. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), pp.971–987.

Oquab, M. et al., 2014. Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, pp. 1717–1724.

Otsu, N., 1979. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1), pp.62–66.

Pan, S. et al., 2015. A Discriminative Cascade CNN Model for Offline Handwritten Digit Recognition. In *14th IAPR International Conference on Machine Vision Applications (MVA)*. Tokyo, Japan, pp. 501–504.

Pan, W.M., Suen, C.Y. & Bui, T.D., 2005. Script Identification using Steerable Gabor Filters. In *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*. Seoul, South Korea: IEEE, p. 883–887 Vol. 2.

Parvez, M.T. & Mahmoud, S.A., 2013. Offline Arabic Handwritten Text Recognition: A Survey. *ACM Computing Surveys*, 45(2), pp.1–35.

Pastor, A.G., 2014. *Bernoulli HMMs for Handwritten Text Recognition*. Ph.D. Thesis, Polytechnic University of Valencia, Spain.

Pechwitz, M. et al., 2002. IFN/ENIT -Database of Handwritten Arabic Words. In *Colloque International Francophone sur l'Écrit et le Document*. Friborg, Switzerland, pp. 129–136.

Pechwitz, M. & Maergner, V., 2003. HMM based approach for handwritten arabic word recognition using the IFN/ENIT - database. In *Seventh International Conference on Document Analysis and Recognition (ICDAR '03)*. Washington, DC, USA: IEEE Comput. Soc, pp. 890–894.

Perronnin, F., Sánchez, J. & Mensink, T., 2010. Improving the Fisher Kernel for Large-scale Image Classification. In *11th European conference on computer vision (ECCV 2010)*. Crete, Greece: Springer Berlin Heidelberg, pp. 143–156.

Pham, V. et al., 2014. Dropout Improves Recurrent Neural Networks for Handwriting Recognition. In *14th International Conference on Frontiers in Handwriting Recognition*. Crete, Greece: IEEE, pp. 285–290.

Philbin, J. et al., 2007. Object Retrieval with Large Vocabularies and Fast Spatial Matching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*. Minneapolis, MN, USA: IEEE, pp. 1–8.

Poggio, T. & Riesenhuber, M., 1999. Hierarchical Models of Object Recognition in Cortex. *Nature Neuroscience*, 2(11), pp.1019–1025.

Porwal, U., Zhou, Y. & Govindaraju, V., 2012. Handwritten Arabic Text Recognition using Deep Belief Networks. In *21st International Conference on Pattern Recognition (ICPR)*. Tsukuba, JAPAN: IEEE, pp. 302–305.

Raina, R. et al., 2007. Self-Taught Learning: Transfer Learning from Unlabeled Data. In

24th international conference on on Machine learning. Corvallis, OR, USA: ACM, pp. 759–766.

Rajput, G.G. & Anita, H.B., 2011. Handwritten Script Identification from a Bi-Script Document at Line Level using Gabor Filters. In *International Workshop on Soft Computing Applications and Knowledge Discovery*. Moscow, Russia, pp. 94–101.

Romero, V., Giménez, A. & Juan, A., 2007. Explicit Modelling of Invariances in Bernoulli Mixtures for Binary Images. In *3rd Iberian conference on Pattern Recognition and Image Analysis (IBPRIA '07)*. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 539–546.

Rothacker, L., Fink, G.A., et al., 2013. Bag-of-Features HMMs for Segmentation-Free Bangla Word Spotting. In *4th International Workshop on Multilingual OCR (MOCR '13)*. Washington, DC, USA: ACM Press, pp. 1–5.

Rothacker, L., 2011. *Learning Bag-of-Features Representations for Handwriting Recognition*. Diploma thesis, Technische Universität Dortmund, Germany.

Rothacker, L. & Fink, G.A., 2015. Segmentation-Free Query-by-String Word Spotting with Bag-of-Features HMMs. In *13th International Conference on Document Analysis and Recognition (ICDAR)*. Tunis, Tunisia: IEEE, pp. 661–665.

Rothacker, L., Rusiñol, M. & Fink, G.A., 2013. Bag-of-Features HMMs for Segmentation-Free Word Spotting in Handwritten Documents. In *2013 12th International Conference on Document Analysis and Recognition (ICDAR)*. Washington, DC, USA, pp. 1305–1309.

Rothacker, L., Vajda, S. & Fink, G. a., 2012. Bag-of-Features Representations for Offline Handwriting Recognition Applied to Arabic Script. In *International Conference on Frontiers in Handwriting Recognition*. Bari, Italy: IEEE, pp. 149–154.

Rusiñol, M. et al., 2011. Browsing Heterogeneous Document Collections by a Segmentation-Free Word Spotting Method. In *11th International Conference on Document Analysis and Recognition (ICDAR 2011)*. Beijing, China: IEEE, pp. 63–67.

Rusiñol, M. et al., 2015. Efficient Segmentation-Free Keyword Spotting in Historical Document Collections. *Pattern Recognition*, 48(2), pp.545–555.

Said, H.E.S., Tan, T.N. & Baker, K.D., 2000. Personal Identification Based on Handwriting. *Pattern Recognition*, 33(1), pp.149–160.

Salakhutdinov, R. & Hinton, G., 2007. Learning a Nonlinear Embedding by Preserving Class Neighbourhood Structure. *AI and Statistics*, 3(1), pp.412–419.

Sánchez, J. et al., 2013. Image Classification with the Fisher Vector: Theory and Practice. *International Journal of Computer Vision*, 105(3), pp.222–245.

Schmidhuber, J., 2015. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61, pp.85–117.

Shekhar, R. & Jawahar, C.V., 2013. Document Specific Sparse Coding for Word Retrieval. In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA, pp. 643–647.

Shekhar, R. & Jawahar, C.V., 2012. Word Image Retrieval Using Bag of Visual Words. In *2012 10th IAPR International Workshop on Document Analysis Systems*. Queensland, Australia: IEEE, pp. 297–301.

Simard, P., Steinkraus, D. & Platt, J.C.C., 2003. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. In *Seventh International Conference on Document Analysis and Recognition (ICDAR '03)*. Washington, DC, USA: IEEE Comput. Soc, pp. 958–963.

Simonyan, K., Vedaldi, A. & Zisserman, A., 2013. Deep Fisher Networks for Large-Scale Image Classification. In C. J. C. Burges et al., eds. *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., pp. 163–171.

Sivic, J. & Zisserman, A., 2003. Video Google: A Text Retrieval Approach to Object Matching in Videos. In *Ninth IEEE International Conference on Computer Vision*. Nice, France: IEEE, pp. 1470–1477.

Soman, S.T., Nandigam, A. & Chakravarthy, V.S., 2013. An Efficient Multiclassifier System Based on Convolutional Neural Network for Offline Handwritten Telugu Character Recognition. In *19th National Conference on Communications (NCC 2013)*. New Delhi, India: IEEE, pp. 1–5.

Stahlberg, F. & Vogel, S., 2015. The QCRI Recognition System for Handwritten Arabic. In *International Conference on Image Analysis and Processing (ICIAP 2015)*. Genova, Italy: Springer International Publishing, pp. 276–286.

Sudholt, S., Rothacker, L. & Fink, G.A., 2015. Learning Local Image Descriptors for Word Spotting. In *13th International Conference on Document Analysis and Recognition (ICDAR)*. Tunis, Tunisia: IEEE, pp. 651–655.

Szegedy, C. et al., 2015. Going Deeper With Convolutions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA, pp. 1–9.

Szeliski, R., 2011. *Computer Vision: Algorithms and Applications*, Springer.

Tencer, L., Renakova, M. & Cheriet, M., 2013. Sketch-Based Retrieval of Document

Illustrations and Regions of Interest. In *12th International Conference on Document Analysis and Recognition (ICDAR 2013)*. Washington, DC, USA, pp. 728–732.

Tipping, M.E. & Bishop, C.M., 1999. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society*, 61(3), pp.611–622.

TransLectures-UPV-Team, 2014. TLK: The TransLectures-UPV Toolkit - tLrecognise manual page. Available at: https://www.translectures.eu//doctools/manpages/tLrecognise.1.html [Accessed November 22, 2016].

Tuytelaars, T. & Mikolajczyk, K., 2007. Local Invariant Feature Detectors: A Survey. *Foundations and Trends in Computer Graphics and Vision*, 3, pp.177–280.

Vedaldi, A. & Fulkerson, B., 2010. VLFeat: An Open and Portable Library of Computer Vision Algorithms. In *18th ACM international conference on Multimedia*. Firenze, Italy: ACM Press, pp. 1469–1472.

Wang, J. et al., 2010. Locality-Constrained Linear Coding for image classification. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA USA: IEEE, pp. 3360–3367.

Wang, T. et al., 2012. End-to-End Text Recognition with Convolutional Neural Networks. In *21st International Conference on Pattern Recognition (ICPR)*. Tsukuba, JAPAN, pp. 3304–3308.

Weng, J., Ahuja, N. & Huang, T.S., 1992. Cresceptron: A Self-Organizing Neural Network which Grows Pdaptively. In *IJCNN International Joint Conference on Neural Networks*. Beijing, China: IEEE, pp. 576–581.

Xu, R. & Wunsch, D., 2005. Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks*, 16(3), pp.645–78.

Yang, X. & Cheng, K.T.T., 2014. Local Difference Binary for Ultrafast and Distinctive Feature Description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1), pp.188–194.

Young, I.T., Gerbrands, J.J. & Van Vliet, L.J., 1998. *Fundamentals of Image Processing*, Delft, The Netherlands: Delft University of Technology.

Young, S. et al., 2006. *The HTK Book (for HTK Version 3.4)*, Cambridge University Engineering Department.

Zaafouri, A., Sayadi, M. & Fnaiech, F., 2015. A Vision Approach for Expiry Date Recognition using Stretched Gabor Features. *The International Arab Journal of Information Technology*, 12(5), pp.448–455.

Zagoris, K. et al., 2014. Distinction Between Handwritten and Machine-Printed Text Based on the Bag of Visual Words Model. *Pattern Recognition*, 47(3), pp.1051–1062.

Zhang, J. et al., 2007. Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study. *International Journal of Computer Vision*, 73(2), pp.213–238.

Zhang, Y. et al., 2014. A Novel Biologically Inspired Local Feature Descriptor. *Biological Cybernetics*, 108(3), pp.275–290.

Zhou, X. et al., 2010. Image Classification Using Super-Vector Coding of Local Image Descriptors. In K. Daniilidis, P. Maragos, & N. Paragios, eds. *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision*. Lecture Notes in Computer Science. Crete, Greece: Springer Berlin Heidelberg, pp. 141–154.

VITA

Name: Mohammed Omer Haj Assayony

Nationality: Yemeni

Date of Birth: Mar 04, 1980

Email: m.assayony@gmail.com

Academic Background:

Sep. 1999 - Jul. 2003: B.Sc. (Computer Science), Al-Ahgaff University, Yemen.

Sep. 2003 – Aug-2004: Teacher, Modern International Institute, Yemen.

Sep. 2004 – Jun. 2007: Junior lecturer, Alandalus University, Yemen.

Jul. 2007 - Dec. 2008: MS (Computer Science), Universiti Sains Malaysia, Malaysia.

Jan 2009 – Aug 2011: Lecturer, Alandalus University, Yemen.

Sep. 2011 – Jan 2017: Ph.D. (Computer Science and Engineering), King Fahd University of Petroleum and Minerals, Saudi Arabia